Technical Report

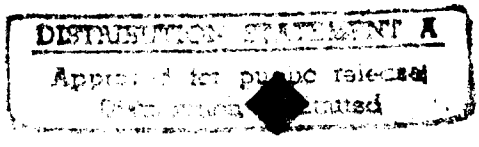
CMU/SEI-91-TR-28
ESD-TR-91-28

Carnegie Mellon University
Software Engineering Institute

Application of Feature-Oriented Domain Analysis to the Army Movement Control Domain

Sholom G. Cohen
Jay L. Stanley, Jr.
A. Spencer Peterson
Robert W. Krut, Jr.

June 1992

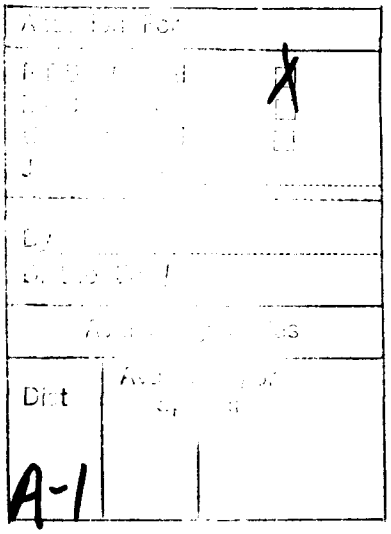


Technical Report
CMU/SEI-91-TR-28
ESD-TR-91-28
June 1992

Application of Feature-Oriented Domain Analysis to the Army Movement Control Domain

Sholom G. Cohen
Jay L. Stanley, Jr.
A. Spencer Peterson
Robert W. Krut, Jr.

Feature-Oriented Domain Analysis Project



Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

This technical report was prepared for the

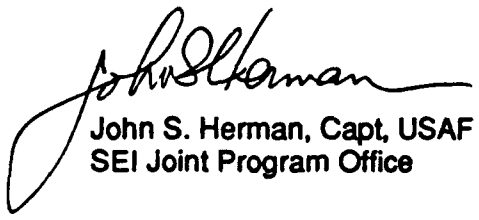
SEI Joint Program Office
ESD/AVS
Hanscom AFB, MA 01731

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

Review and Approval

This report has been reviewed and is approved for publication.

FOR THE COMMANDER



John S. Herman, Capt, USAF
SEI Joint Program Office

The Software Engineering Institute is sponsored by the U.S. Department of Defense.
This report was funded by the U.S. Department of Defense.

Table of Contents

List Of Figures

List Of Tables

Application of Feature-Oriented Domain Analysis to the Army Movement Control Domain

Abstract: This report documents an analysis of the army movement control domain performed by the Software Engineering Institute (SEI) and a team of movement control experts from the Army. This report includes common terminology and requirements extracted from Army doctrine, experts in the field, and movement control systems. The report also describes the potential for prototyping of systems using domain analysis products and the tool support needed.

1. Introduction

Movement control plays a major role in the delivery and sustainment of combat forces.The successful deployment of these forces often determines the outcome of campaigns, major operations, battles, and engagements. Movement control has been defined in [FM 55-10] as:

> *"The planning, routing, scheduling, control, coordination, and in-transit visibility of personnel, units, equipment, and supplies moving over lines of communication and the commitment of allocated transportation assets according to command planning directives."*

Therefore, movement control involves synchronizing and integrating logistics, movement information, and programs that span the three levels of war: strategic, operational, and tactical. Movement control balances requirements against capabilities and allocates resources based on the combat commander's priorities.

1.1. Audience for this Report

The information contained in this document is organized in a manner that provides clear insight into the movement control domain. Its intended audience need not be experts in the area of movement control. The report addresses the needs of a variety of readers and:

- Is intended for developers interested in the requirements for army movement control application software.
- Provides an understanding of the application of a specific domain analysis method.
- Is directed towards individuals generally interested in the application of a domain analysis method.
- Provides an example of the application of a specific domain analysis method in the command and control area.

Although the movement control analysis spanned strategic, operational, and tactical levels of warfare, the primary focus was on the operational and tactical levels. More specifically, the products resulting from the domain analysis will be targeted for users at the corps level of command and below. This is because the organizations participating and showing significant interest in this work are working in these focus areas with the movement control domain.

1.2. Purpose of Analysis

The products resulting from the domain analysis provide a common basis for communicating user needs to implementors of movement control applications. In addition to providing domain analysis products, the purpose of the analysis was also to validate the SEI Feature-Oriented Domain Analysis method (FODA) presented in [SEI90a] and provide an approach for future domain analysis.

1.2.1. Develop Products to Support Software Implementation

The Army Tactical Command and Control System (ATCCS) consists of five battlefield functional areas (BFAs). The BFAs are Air Defense, Combat Service Support, Field Artillery, Intelligence, and Maneuver Control. These BFAs share common application requirements such as movement control. The domain analysis was intended to produce products that support the implementation of related movement control software applications within the BFAs and other army systems.

1.2.2. Validating the Method

During the feasibility study, FODA was applied to the window manager domain. The window manager domain was scoped for limited size and selected because it was well documented. The next step in validating the method was to apply FODA to a more challenging domain. The ATCCS movement control domain was selected as representative of a larger, more complex, and less well-documented domain.

1.2.3. Provide Approach for Using FODA in Other Domains

Following successful application of FODA within the movement control domain, the method can be applied to other software application domains. These applications may be used for ATCCS and other Army projects. Domain analysis has the potential for saving time, effort, and expense by promoting effective software reuse. Therefore, the FODA methodology will serve long-term reuse objectives.

1.3. Report Overview

This report summarizes the method and the sources used during the analysis. The report provides a summary of the products that are given in detail in the appendices (bound as a separate volume). The report contains a description of the use of the products in building a move-

ment control application, as well as a summary of lessons learned. The following is a summary of the remaining sections of this report:

Section 2: Overview of the Method - provides a short review of the FODA method. This review concentrates on how the method was applied during the first two phases of the movement control domain analysis and how we plan to perform the third and final phase of the method for this domain.

Section 3: Technical Approach - provides insight on information gathered from domain experts, documentation, and systems. Other topics addressed within this section are tools used to aid model development, special training, and results of the workshop conducted for the purpose of gathering domain requirements and validating interim results.

Section 4: Movement Control Domain Analysis Review - provides a concise view of the products of the domain analysis and discuss how each adds to an understanding of the movement control domain. This section also describes validation of the model against existing systems.

Section 5: Application of the Domain Model in System Development - shows an extended example of how the domain model products are used in the construction, usage, and validation of a prototying tool for modeling movement control systems.

Section 6: Conclusion - addresses two primary topics: lessons learned and recommendations for future research. Lessons learned addresses pertinent information that will be used to refine the methodology. Recommendations for future research focuses on future directions for the current project.

2. Overview of Feature-Oriented Domain Analysis Method

The SEI domain analysis of movement control identified, collected, organized, and represented the relevant information in the movement control domain. This analysis studied existing systems and their development histories, underlying theory, emerging technology within the domain, and captured knowledge from domain experts. In performing this domain analysis, the SEI used the FODA method. This method supports the discovery, analysis, and documentation of commonality and differences within a domain.

FODA defines a process for domain analysis and establishes specific products for later use. Three basic phases characterize the FODA process:

1. *Context Analysis*: defining the extent (or bounds) of a domain for analysis.
2. *Domain Modeling*: providing a description of the problem space in the domain that is addressed by software.
3. *Architecture Modeling*: creating the software architecture(s) for implementing solutions to the problems in the domain.

The application of the FODA during each of the phases is described in detail in Sections 2.1-2.3.

During the analysis there are a variety of participants who must provide information, develop domain analysis products, and review the results. The FODA report describes three groups of participants in the domain analysis process. They are the:

1. *Sources*. These participants provide information needed during the analysis. They may be further characterized as:
   - *End users*. The personnel that use systems in the domain. These participants know how the systems they use operate, understand where the systems fit in a larger flow of control, and what capabilities are missing from current systems.
   - *Domain experts*. The personnel that provide information about systems in the domain.
2. *Producers*. These participants gather the information, perform the analysis, and produce the products. They are familiar with the FODA method and applying it to gather and organize domain knowledge.
3. *Consumers* - These participants use the domain analysis products. They may be further characterized as:
   - *End users*. The same as in *Sources* above.
   - *Requirements analysts*. The specifiers of new systems in the domain.

- *Software designers.* The personnel designing new systems in the domain.

The participants in an analysis may not play unique roles, i.e., the requirements analyst who has worked on the specifications of several systems in the domain may be one of the domain experts. There is a clear distinction, however, among the roles of consumers of domain analysis. The role categories of consumers map easily onto the roles of various Army organizations that take part in the software acquisition process [ACAM86]. This process is affected by the interactions among the organizations performing the domain analysis and the users of the analysis. Two important organization types that are part of the acquisition process are:

1. *Combat Developer.* This is the Army term for the user representative in the acquisitions process. The combat developer is a key source of domain expertise, as he is familiar with requirements for a related set of software systems. Capturing commonality at this level is critical in forming the domain model. The combat developer also is a user of the domain model, utilizing the model as the basis for requirements elicitation and specification.
2. *Material Developer.* The material developer works with the combat developer in identifying and specifying system requirements. The material developer has two primary roles:
    a. *Acquisition agent.* Generally, this role is performed by the Program Executive Officer (PEO) and specific Product Managers (PMs) for individual systems. For development of reusable software, the material developer will serve as the key acquisition agent and must be an active participant in the domain analysis.
    b. *Implementor.* The contractor or an internal development organization is the actual developer of the domain model and architecture. The model developer or a separate implementor may also be users of the models, building a system based on the models or using all or parts of the models in system development, reusable component development, and training.

An overview of each of the phases within the FODA process and the relationships between their products and their consumers is given in the following sections. This discussion of FODA is given in light of its application to the movement control domain.

2.1. Context Analysis

Context analysis defines the scope of a domain that is likely to yield useful domain products. During the context analysis for this domain, the relationships between the movement control domain and the elements external to it were established and analyzed for variability. The kinds of variability to be accounted for are, for example, when applications in the domain have different data requirements and/or operating environments. The results of the context analysis, along with other factors such as availability of domain expertise, domain data, and project constraints, were used to limit the scope the domain. Sections 3.1and 3.2 describe the information

resources used to perform our analysis of the movement control domain. The context analysis for the movement control domain is documented in detail in a previous SEI report [SEI91a].

The product resulting from the context analysis is the *context model.* This model includes a structure diagram and a context diagram. The structure diagram for this domain is an informal block diagram in which the movement control domain is placed relative to higher, lower, and peer-level domains, all within a general view of the domain's applicability. Higher level domains are those of which the domain under analysis is a part of or to which it applies. For example, movement control is only one of several domains in the Army command and control domain. Lower level domains (or subdomains) are those within the scope of the domain under analysis, but which are well understood. Examples of these lower level domains for movement control are User Interfaces and Common Message Services. Any other relevant domains (i.e., peer domains) must also be included in the diagram.

The movement control context diagram is a data flow diagram showing data flows between a generalized application within the domain and the other entities and abstractions with which it communicates. One thing that differentiates the use of data flow diagrams in domain analysis from other typical uses is that the variability of the data flows across the domain boundary must be accounted for with either a set of diagrams or text describing the differences.

These products provide the domain analysis participants, mentioned at the beginning of this section, with a common understanding of:

- The scope of the domain
- The inputs/outputs
- Stored data requirements (at a high level) for the domain

Section 4.2 discusses the products of the context analysis and their use in more detail.

2.2. Domain Modeling

Domain modeling identifies the commonalities and differences that characterize the applications within the domain. The domain model documented in Section 4.3 of this report applies to the scope of movement control established during the context analysis and refined in Section 4.1. The domain modeling phase consist of three major activities. A brief description of each activity and its results is given below.

1. *Entity-Relationship* (ER) *Modeling* captures and defines the domain knowledge and data requirements that are essential for implementing applications in the domain. The movement control domain is rich in data requirements including data to characterize unit and orders information (see Section 4.3.1). Domain knowledge typically is information that is deeply embedded in the software and is often difficult to trace. Those who maintain or reuse software need this information in order to understand the problems the domain addresses.

The ER model is used primarily by the requirements analyst and the software designer to ensure that the proper data abstractions and decompositions are used in the development of the system. The ER model also defines data that is assumed to come from external sources.

2. *Feature Analysis* captures the end user's understanding of the general capabilities of applications in a domain. For the movement control domain, the commonalities and differences of interest to end users among related movement systems were designated as features and are depicted in the *feature model.* These features, which describe the context of domain applications, the needed operations and their attributes, and representation variations are important results because the feature model generalizes and parameterizes the other models produced in this domain analysis.

   The feature model is the chief means of communication between the end users (in movement control, the combat developer organizations) and the developers (i.e., the material developers and implementor) of new applications. The movement features are meaningful to the end users and can assist the requirements analysts in the derivation of a system specification that will provide the desired movement control capabilities. Previously, combat developers have had difficulty specifying their needs. The feature model provides them with a complete and consistent view of the movement control domain. The combat developer will select features and the requirements analyst can validate them for completeness and consistency. The domain modeling tool (see Section 3.4.1) allows prototyping of the selected features by the software designer during the software development process for a new system in the domain.

3. *Functional Analysis* identifies the control and data flow commonalities and differences of the applications in a domain. This activity abstracts and then structures the common functions found in the domain and the sequencing of those actions into a model. Common features and ER model entities form the basis for the abstract functional model. The control and data flow of an individual application can be instantiated or derived from the functional model with appropriate adaptation. The functional model for the movement control domain is described in detail in Section 4.3.3.

   The functional model is the foundation upon which the software designer begins the process of understanding how to provide the features and make use of the entities selected.

The domain modeling process also produces an extensive *Domain Dictionary* (Appendix G) of terms and/or abbreviations that are used in describing the features and entities in movement control, and a textual description of the features and entities themselves. If multiple terms are used to convey an equivalent concept, each is listed in the dictionary with the most frequently used term identified as the primary term with the needed definitions(s) and the other terms referring back to the primary.

The domain dictionary has been found to be one of the most useful products of a domain analysis. The dictionary helps to alleviate a great deal of miscommunication because it provides the users of the domain information with a central location to look for terms and abbreviations

that are completely new to them, or for definitions of terms that are used differently or in a very specific way within the domain.

The resources used during the domain modeling phase of this analysis are described further in Chapter 3. The products of this effort, i.e., the ER model, feature model, functional model, and domain dictionary are included in various sections of Chapter 4 and/or as appendices to this report.

2.3. Architectural Modeling

Architectural modeling provides a software solution for applications in the domain. An architectural model (also known as a design reference model) is developed in this phase and detailed design and component construction can be done from this model. This architectural model is a high-level design for applications in a domain. It focuses on identifying concurrent processes and domain-oriented common modules, and on allocating the features, functions, and data objects defined in the domain models to the processes and modules.

The FODA report describes the use of DARTS methodology [GOMAA84] for the architectural modeling. For the movement control domain, the architectural model will be based on the *structural modeling* method. This method is based upon work performed by the Software Architecture Engineering (SAE) project at the SEI on projects that have developed software architectures.

In the structural modeling approach, the architecture is based upon the repeated use of a set of software structures. The consistent use of structures throughout the architecture is called the Object Connection Update (OCU) paradigm, documented in [SEI88]. The OCU concepts and techniques have been refined into the structural modeling process, which was initially defined in [SEI90b] and is currently being used on several external projects under the supervision of the SAE Project. A more detailed description of the method is currently being developed by the projects

The OCU paradigm can be described briefly by defining the terms as they are used in connection with the paradigm.

Object A software implementation of a real-world entity, either a physical object or a logical object that is to be treated as if it were real (an organization). Given the attributes of the object and its operational state, an object maps the relevant effects of its environment onto itself. The implementation isolates individual effects and is unaware of its connections to other objects. The implementation of a object may be a composition of other objects.

Connection The mechanism for transferring state information between objects. Processing a connection involves reading the state of some objects on the connection and broadcasting to others.

Update The gating and processing of a connection or other input to derive a new output and/or internal state of an object.

A complete system is comprised of three levels of software component activities:

1. The executive, which handles the coordination of objects, the connections between them, and the ordering of updates.
2. The system, which organizes a set of related objects into a more meaningful whole. It provides internal coordination for its objects and interfaces them to other systems.
3. The object, which encapsulates and manages the information about a single entity, including the attributes and state information.

These three levels map well onto various aspects of the ER and functional models developed during the domain modeling process. The executive component is derived from the functional model and defines the independent activities that share information and synchronize with one another. The systems components are derived from the detailed behavioral states and the activities they control. Finally, the objects are derived from examination of the ER model and the data and operations needed by the activities.

2.4. Applying the Results of Domain Analysis

FODA defines a method for performing domain analysis and describes the products of an analysis. The application of the domain analysis method within the Army domain has helped refine the method and establish ways of applying the products.

Figure 2-1 shows the three components of the domain model: the feature model, the entity relationship model, and the functional model. A combat developer works with the domain analyst and these products define requirements for a system. The three steps in the process are:

1. The combat developer and domain analyst use the feature model as a vehicle for communicating system needs. The domain analyst will turn these needs into a selection of features. In addition, composition rules among features will automatically add specific features to the new system.
2. The domain analyst uses the entity relationship model to explain the *objects* that compose a system. This helps the combat developer understand the data requirements as well as other systems and data structures with which the system must interoperate.
3. The functional model is then used to describe commonality and differences in data and control flow resulting from differing combinations of features.

The product of feature selection is the definition of capabilities of the system under development as shown in Figure 2-1.

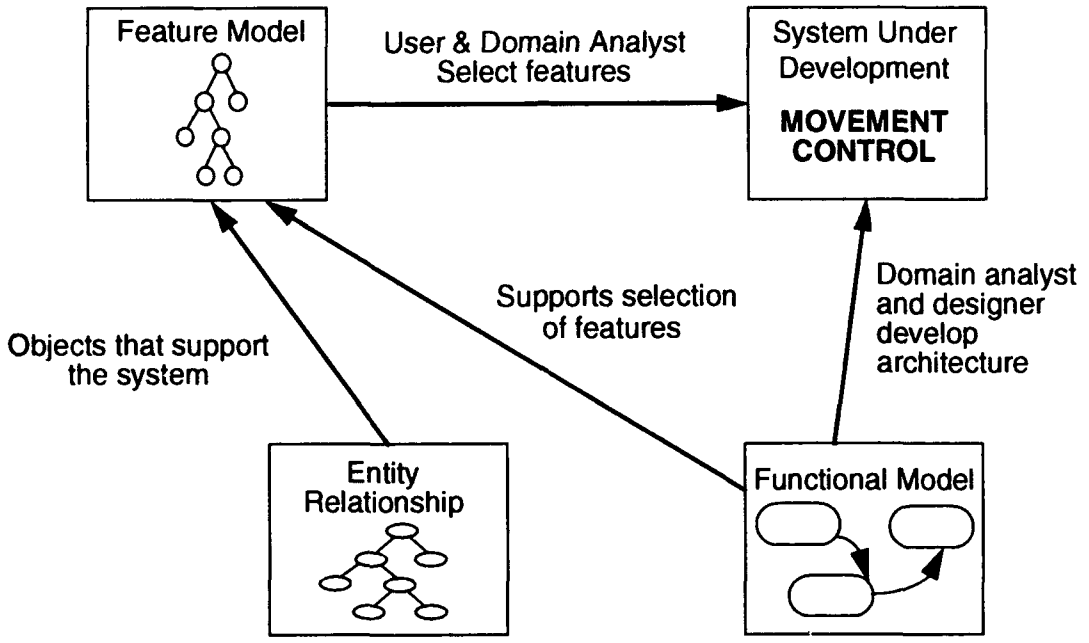


Figure 2-1 Use of Domain Model in System Development

The functional model supports feature selection as well as architectural development. Feature selection will parameterize the functional model, establishing the dynamics of interacting system capabilities. A combat developer will utilize this information in making choices that will affect both system control and operations. For example, a choice of features may affect the sequence of operations or eliminate those operations altogether. Another important aspect of this model is the definition of data flow resulting from these operations. The system dynamics necessary to meet the desired system capabilities may depend on specific feature selections.

When implementing the desired features, the domain analyst and software designer will work jointly to establish the software architecture. The functional model defines data exported by specific activities as well as those required for input by other activities. The model also shows the control necessary to start an activity to effect the data flow. The architectural structure described in Section 2.3 is a direct product of this type of interaction. The detailed realization of all data flows and the control necessary to accomplish them is a key component of software design. Using the feature model, the software designer engineers a general architecture that supports implementation of common features that can also be parameterized for tailorability for meeting optional and alternative features.

This application of the domain analysis products feeds into a life-cycle view of *model-based software development.* This process is illustrated in Figure 2-2. Under this development approach, models exist to help in both *setting* the problem and in *solving* the problem. Domain

analysis and its products are the model base for understanding the user needs and obtaining requirements. Architectural models provide a structure for building a solution. Finally, implementation models provide components, software generator tools, and software composition tools to support production of the deliverable software products. Where the models are inadequate for understanding the problem space and producing a solution, the life cycle must include unprecedented development. In addition to filling the gaps in existing models, unprecedented development will lead to refinement of existing models or to the realization that the existing models are no longer adequate.

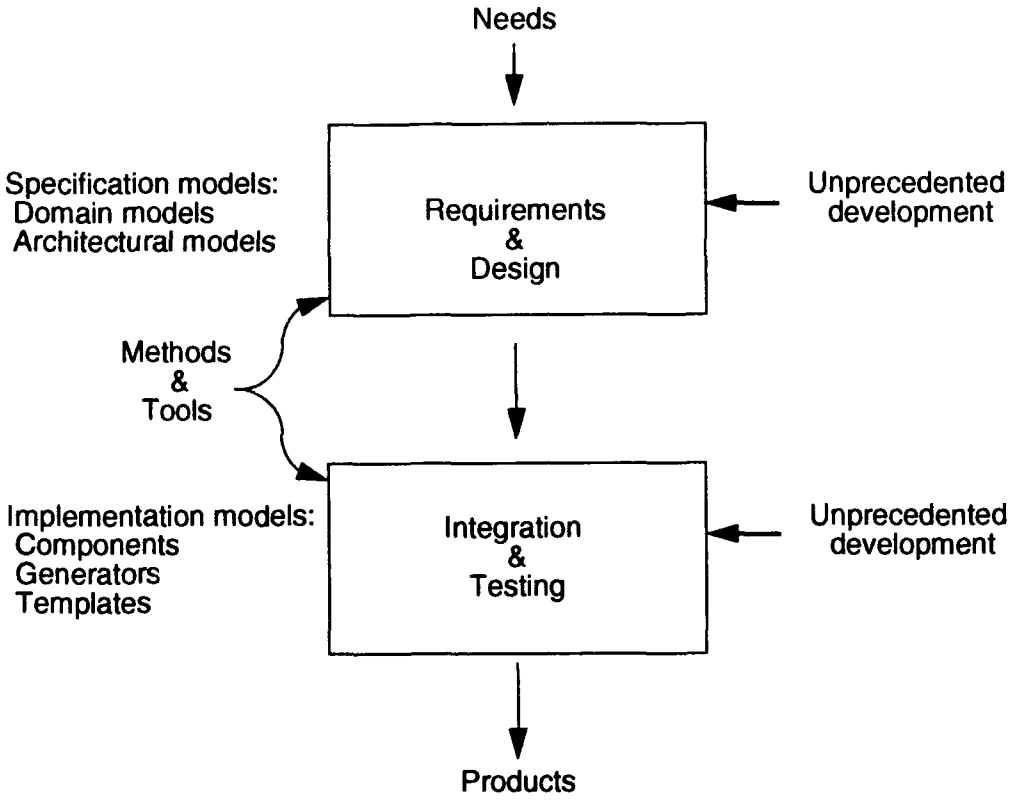


Figure 2-2 Domain Analysis and Model-Based Development

The life-cycle model also shows the importance of methods and tools. While general tool support is necessary for configuration management and other general software engineering activities, specialized model-based tools and methods will support specification and implementation activities. For domain analysis, these tools will provide functionality to:

- Document the domain model
- Support feature selection for model-based specification
- Perform prototyping to animate specifications

Chapter 5 of this report describes the use of a domain analysis tool supporting these functions.

3. Technical Approach

The following sections contain sources and summaries information gathered during the analysis. The section also describes the tools and techniques used by the analyst during the effort.

3.1. Domain Expert Discussion

An important step in domain analysis is understanding the customer (user/developer) community. During the Army Movement Control Domain Analysis Project, the customer community was examined to identify key components of the army acquisition process. This examination revealed three primary components:

1. *Production.* Converts raw material into the intermediate goods required by the combat system. Within the production system, there are two major subcomponents:

   - *Training and Doctrine Command (TRADOC).* Responsibilities of TRADOC are to produce training, doctrine, tactics, and techniques for combat operations. TRADOC is also the user representative in the research, development, and acquisition process of weapon systems. TRADOC conducts all combat developments that are not specifically designated to other Army commands and agencies.

   - *Army Materiel Command (AMC).* Produces weapon systems and other related weapons materiel and supports the systems after fielding. The Army Materiel Command performs assigned materiel and related functions for research and development, development test and evaluation, acquisition, and logistics support of materiel systems as required by the Army.

2. *Combat.* Converts the Army's intermediate products, obtained from the production subsystem, into combat-ready forces.

3. *Integration/Coordination.* Decides what is to be produced or accomplished by the whole system and makes sure that the system performs as expected [ACAM86].

TRADOC and AMC share the responsibility of research, development, and acquisition. Due to this, TRADOC combat developers and AMC materiel developers played a significant role in the development of the domain model.

During the early stages (context analysis phase and domain modeling phase) the thrust of the analysis was identifying system requirements. Therefore, the domain analysis relied heavily on the combat developers and the user representative during materiel acquisition for domain expertise (see Figure 3-1).

As the analysis progresses into the later stages of the domain modeling process, more interest is placed on converting the user requirements to a common software system architecture. The

materiel developers now play a more significant role in the domain analysis. Figure 3-1 illustrates the typical developer involvement during domain analysis.

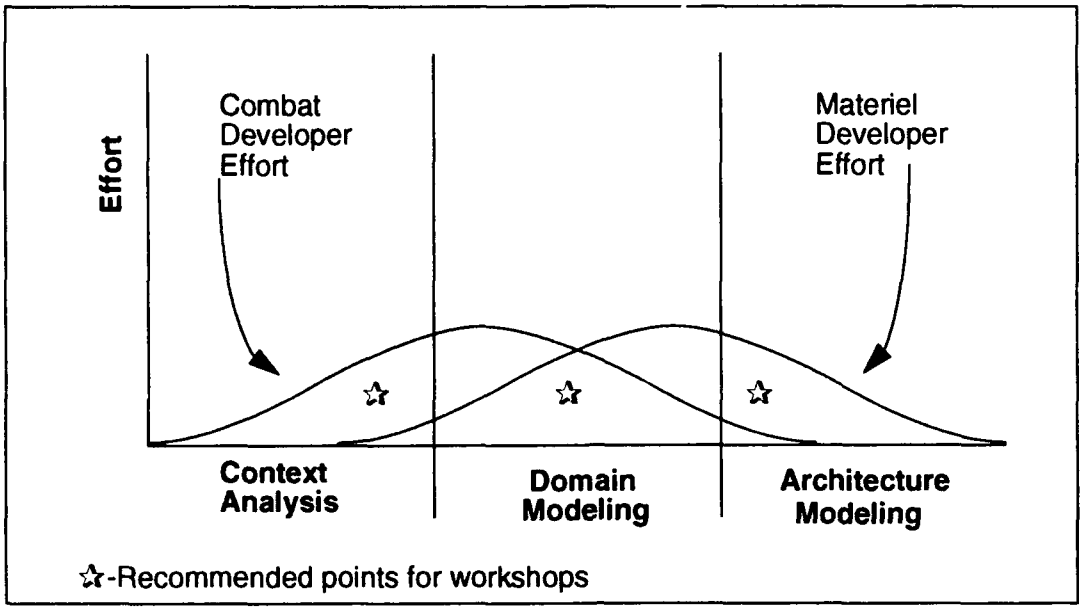


Figure 3-1 Developer Effort During Domain Analysis

The domain workshop is necessary for obtaining domain information and validation of existing domain analysis products. The participants in the workshop will include domain experts from TRADOC and AMC. As illustrated by the figure, there are three appropriate times for conducting a domain analysis workshop:

1. *Prior to the transition into the Domain Modeling Phase.* As a result of the preliminary scoping process, the domain analyst understands enough of the terminology and issues to ask intelligent questions associated with scoping the domain and building the model skeleton.
2. *Midway through the Domain Modeling Phase.* At this point, the most knowledgable and articulate of the combat developer domain experts have been identified, the preliminary domain model is developed, and it can be used as a focal point of feature discussion. Materiel developer domain experts can begin orientation into the domain.
3. *Shortly after the transition to the Architecture Modeling Phase.* Materiel developers understand the issues surrounding the domain, a few of the implementation issues have been discussed, and the first draft of the system architecture has been completed and can be used as the focal point for uncovering unknown implementation problems.

The next section discusses the finding of the workshop conducted during this project.

3.2. Domain Modeling Workshop

An Army movement control workshop brought together experts from the U.S. Army along with domain developers from the SEI, industry, and academia. The workshop proved to be an effective means for:

- Discussions of commonalities among movement control systems.
- New and evolving system requirements.
- Underlying theory of movement control.

The effectiveness of the workshop is dependent on two major factors:

1. *Timing of the Workshop.* The domain analyst should be reasonably familiar with the domain before scheduling the workshop. This timing of the workshop allows:
   - Identification and invitation of the most important domain experts.
   - Review of draft versions of the entity relationship and feature models. During the workshop, the interaction between the different various models and the use of the FODA method could be demonstrated. This aspect of the workshop provides a forum for the domain analysts to articulate their current understanding of the domain. Use of the models also provides a mechanism for the domain experts to establish common terminology, discuss additional features, and identify issues that require further domain analysis.
2. *Control of Discussion.* There are several critical factors which must be considered during open discussions:
   - Because the participants are from diverse backgrounds, each participant understands the system from a slightly different point of view. Control of the discussion requires extracting the commonality that exists between the different points of view. The domain analyst must then restate these characteristics in a manner that can be agreed upon by the experts.
   - The discussion must focus on key characteristics of the domain. Because the participants are experts in the domain, there is a tendency for them to sidetrack and elaborate on the relatively unimportant issues. Knowledge of the domain at this point in the analysis enables the participants to maintain proper focus on the appropriate issues within the domain.

3.3. Sources of Domain Information

The domain analysis relied on domain experts in movement control, systems that performed movement control functions and Army doctrine, for domain information. The following subsections document these sources of information.

3.3.1. Domain Experts Involved with the Project

The domain analysis obtained expert knowledge from the U.S. Army labs, schools, and development organizations. Domain experts are listed in Table 1.

3.3.2. System Discussion

There are relatively few software systems with movement control features. This was one of the primary differences between this analysis and that of the feasibility study [SEI90a]. During the feasibility study, a large percentage of the window manager software features could be viewed directly from the screen. The lack of example systems within the movement control domain increased the work of the domain analyst. Instead of having working systems from which to extract features and other domain information, the analyst was forced to uncover the information from requirements documents, doctrine, and discussions with domain experts.

However, there were a few computer based systems discovered that exhibited limited movement control capability. The vast majority, though, were limited to the convoy planning aspects of movement control. Others were in the early stages of development and were merely prototypes arising from the concept exploration phase of development.

Table 2 contains a list of systems analyzed during the domain analysis process. Table 3 contains a list of the documentation associated with the reviewed systems.

3.3.3. Army Doctrine

Table 4 provides a list of key Army documents that contributed to the understanding of the domain.

Organization	Name	Contribution to the SEI	Benefits from Domain Analysis
U.S. Army Field Artillery School System: AFATDS	• Major John Garhart	• Source for unit movement control requirements • Provided more insight into the AFATDS movement control module	• Exposure to work in related systems (DAMMS-R, ALBE-GIS, etc.) • Clear picture of common movement control requirements
Magnovox Electronic Systems Company System: AFATDS	• Eric Deets • Scott Black • David Burgdorf	• Source for unit movement control requirements • Provided more insight into the AFATDS movement control module	• Exposure to work in related systems • Clear picture of common movement control requirements • Facilitates development of reusable movement control software

Table 1 List of Domain Experts

Organization	Name	Contribution to the SEI	Benefits from Domain Analysis
U.S. Army CECOM System: CASS	• Bruce Gray • Mike Fragale	• Provide information about the ATCCS/CASS layer	• Understanding of the domain analysis methodology • Knowledge of how the technologies can be used in subsequent CASS efforts
Combined Arms Command Future Battle Lab	• Capt. Jim Shufelt	• Provided insight into the new AirLand Operations concept	• Assurance that the systems that will be developed from the domain analysis process will be able to change to meet the new fighting doctrine
Carnegie Mellon University System: TPFDS	• Steven Roth	• Insight into strategic level movement control characteristics • Demonstration of the CMU TIPFDS prototype	• Exposure to the movement control methodology

Table 1 List of Domain Experts - Continued

Organization	Name	Contribution to the SEI	Benefits from Domain Analysis
U.S. Army Combined Arms Support Command System: CSSCS	• Richard H. Silva	• General understanding of the CSSCS requirements	• Understanding of the full Army movement control model • Clear picture of interface issues and information exchanged between CSSCS and DAMMS-R
Carnegie Group, Inc. System: KBLPS	• Ivan Johnson • Vivien Robinson • Bruno Levy • Victor Saks	• General understanding of the KBLPS	• Provided requirements for the transportation aspects of the KBLPS prototype
U.S. Army Waterways Experiment Station System: ALBE-GIS	• David Horner	• Provided understanding about the GIS for ATCCS	• Provided insight into the Tactical Decision Aids that will be built into the ALBE-GIS
U.S. Army Transportation School	• Capt. Mike Boyle	• Provide information on current practices in movement control • Provide lessons learned from Desert Storm	• Opened up communication channels between Transportation School and developers of DAMMS-R

Table 1 List of Domain Experts - Continued

Organization	Name	Contribution to the SEI	Benefits from Domain Analysis
U.S. Army Combined Arms Support Command System: DAMMS-R	• Tom Snodgrass • William L. Cope	• Source for logistical and unit movement control requirements for DAMMS-R	• Exposure to other movement control systems and GIS alternatives • Clear understanding of the movement control domain requirements which will be used in the development of DAMMS-R
COMARCO/IBS System: DAMMS-R	• William H. Anckaitis • Christopher R. Valentine	• Source for logistical and unit movement control requirements	• Exposure to work in related systems • Broader perspective on Army movement control domain requirements • Facilitation of the development of clear and concise Army requirements documents • Expose issues of typical user interface
Loral Command and Control Systems System: MCS	• Jim Frahm • Leonard Courson • Jim Standlee • James Aucoin	• Provide insight into MCS functionality	• Exposure to work in related systems

Table 1 List of Domain Experts - Continued

System	Description
Department of the Army Movement Management System - Redesign DAMMS-R	The overall objective of DAMMS-R is to provide a reliable, survivable, and responsive information processing capability in support of physical distribution management, movements control, transportation operations, and transportation services in overseas theaters in peace or war.These objectives include capabilities for movement programming, *highway traffic regulating*, cargo forecasting, shipment tracing, holding and diversion, in-transit visibility, fleet and intermodal asset accountability, *monitoring and controlling* of containers, *unit and other specified movements*, load planning, *convoy planning*, and *recording PDN characteristics*.
Advanced Field Artillery Tactical Data System AFATDS	The Movement Contrc subsystem within AFATDS consists of three major functional categc es: *Internal FA Coordination*, which include the initiation of movement requirements and determining the time, location, and routes for a move (*convoy planning*); *Movement Request*, which includes preparation of movement request and reports and resolution of route conflicts *(highway regulation)*; and *External FS Coordination*, which validates and coordinates movement requests between AFATDS and MCS.
Transportation Coordinator Automated Command and Control Information Management System TCACCIS	TCACCIS provides individual units with a *convoy planning* tool for use by reserve units within the continental U.S. It provides some support for *highway regulation* but lacks a deconfliction capability.

Table 2 List of Systems

System	Description
Condensed Army Mobility Module System CAMMS	CAMMS is a software package designed to make mobility predictions of various kinds for use in tactical applications. It contains functionality to provide the following capabilities that are movement control related: current and historical weather effects, cross-country mobility predictions, *on-road mobility predictions*, foot soldier mobility predictions, maneuver damage predictions, *interactive route evaluation*, and *route network evaluation.*
Mobilization Movement Control System MOBCON	MOBCON provides various movement control capabilities for use in the planning and execution of Active Army or Army Reserve unit movements within the U.S. at the state or national level. It provides the following movement control capabilities: *convoy planning* (units must submit movement input data via paper forms) and *highway regulation* (route deconfliction).

Table 2 List of Systems - Continued

Document	Description
ATCCS System Developement Master Plan, Annex A - Current Configuration	This document describes the current configuration of the C^2 systems for each of the five BFAs in ATCCS. It also addresses the CHS and communications programs.
ATCCS System Specification	This document establishes the performance, design, development, and test requirements for ATCCS.
Common ATCCS Support Software (CASS) System/Segment Specification	This document establishes the functional, performance, security, and verification requirements for the CASS.
Segment Specification for the Maneuver Control System (MCS), Segment 11	This document establishes the requirements for Segment 11 of the Maneuver Control System (MCS.)
Computer Program Development Spec. for the Movement Control CPCI, AFATDS	This document establishes the requirements for design, test, performance, and qualification of the computer program identified as the MC CPCI of AFATDS.
Combat Service Support Control System (CSSCS), Segment 3, Functional Reqs.	This document present the requisite functional capabilities of Segment 3 of CSSCS and serves as the basis for Segment 3 design and development.

Table 3 List of System Documents

Document	Description
Functional Description for DAMMS-R, Volume 1	This document is written to provide system requirements to be satisfied, information on performance requirements, and a basis for system testing and maintenance.
DAMMS-R Comparability Analysis Study	This document reflects the result of evaluating candidate systems (fielded or under development) which appeared to meet requirements of 4 subsystems within DAMMS-R.

Table 3 List of System Documents - Continue

Document	Description
FM 5-33: Terrain Analysis	This field manual prescribes basic doctrine and is intended to serve as a primary source of the most current information on terrain analysis procedures.
FM 5-36: Route Reconnaissance and Classification	This field manual describes the fundamentals of route reconnaissance and methods for reconnoitering and classifying routes for military use.
FM 6-20: Fire Support in the AirLand Battle	This field manual establishes the principles of fire support and describes the fire support system in terms of its major components, functions, and required products.
FM 6-20-1: Field Artillery Cannon Battalion	This field manual describes how cannon battalions are organized and how they operate to support the combined arms team.
FM-34-1: Intelligence and Electronics Warfare Operations	This manual expands doctrine contained in FM 100-5. It delineates the IEW misson, the role of IEW in combat, and the principles governing its operations and sustainment.
FM 34-3: Intelligence Analysis	This manual describes the processes, procedures, and techniques used to produce intelligence. It focuses on intelligence production at corp level and below.

Table 4 List of Field Manuals

Document	Description
FM 35-25: Corps Intelligence and Electronic Warfare Operations	This manual provides Army doctrine for corps IEW operations and presents tactics and techniques for accomplishing the four IEW tasks.
FM 55-2: Division Transportation Operations	This manual provides doctrinal guidance concerning organization and functions of division transportation operations in an oversea wartime environment.
FM 55-10: Movement Control in a Theater of Operations	This manual shows how transportation movement resources or modes are managed and controlled in an oversea theater.
FM 55-15: Transportation Reference Data	This manual includes the characteristics of transportation equipment and facilities, and methods for estimating capabilities and requirements for movement.
FM 55-30: Army Motor Transport Units and Operations	This manual is devoted to the movement of cargo by trucks. It covers applicable command and task units. It explains the different types and methods of hauls.
FM 55-60: Army Terminal Operations	This manual is directed to Army terminal operations in a theater of operations. It contains procedures and techniques for planning and executing these operations.
FM 100-5: Operations	This is the Army's keystone warfighting manual. It explains how Army forces plan and conduct campaigns, major operations, and engagements.

Table 4 List of Field Manuals - Continued

Document	Description
FM 100-10: Combat Service Support	This manual depicts the Army CSS organizations and describes how they support commanders by manning, arming, fueling, fixing, and moving their forces.
FM 100-15: Corps Operations	This manual addresses the conduct of corps combat operations and the integration and coordination of combat units and the various support elements.
FM 101-5: Staff Organizations and Operations	This manual prescribes basic doctrine for staff organization and operations. It contains information about the content and format of combat plans and orders.
FM 101-5-1: Operational Terms and Symbols	This manual is the dictionary of operational terms and military symbols. Definitions and terms here agree with those in JCS Pub. 1 and AR 310-25.
FM 101-10-1: Staff Officers' Field Manual - Organizational, Technical & Logistic Data	This manual contains the Table of Organization and Equipment for various kinds of units. It also has data for supply usage and movement planning for these units.
AR 310-25: Dictionary of United States Army Terms	This publication, which defines Army terminlogy, is a supplement to JCS Pub. 1 and contains defintions not found in JCS Pub. 1.
JCS Pub. 1: Department of Defense Dictionary of Military and Associated Terms	This publication supplements standard English language dictionaries with a source of standard terminology for military use.

Table 4 List of Field Manuals - Continued

3.4. Tool Support for Domain Analysis

The FODA feasibility study [SEI90a] determined the need for tools to support both the process of domain analysis and the process by which the products of the domain analysis support software development. The initial intention of the feasibility study was to perform the analysis using manual techniques. As the amount of information needed to describe the domain grew, the manual technique became more complex. To handle the volume and complexity of information gathered during the domain analysis, a set of manual and independent semiautomated methods were used. These tools provided:

- Cross-checking and consistency
- Reusability of data

Representing the results of a domain analysis process is primarily a task of representing a large amount of knowledge. The domain analyst should provide facilities so that the user can access that knowledge quickly and easily. The goal of domain analysis tool support should be to offer an integrated environment for collecting and retrieving the domain model and architectures. The set of manual and independent semiautomated methods used during the feasibility study did not meet that goal. Therefore, the FODA feasibility study recommended that subsequent domain analysis studies investigate integrating tool support into the domain analysis method and produce requirements for specific domain analysis support tools. This recommendation was addressed during the application of FODA to the Army Movement Control Domain.

The following section discusses the tool currently being examined during the Movement Control Domain Analysis. The tool is 001™, (written "zero zero one," but pronounced "double-oh one") created by Hamilton Technologies, Inc. (HTI) [HAMIL86], [MURPH90]. 001 provides the modeler the ability to:

- Integrate entities, features, and functions into a consistent model.
- Map selected features to a system under development.
- Generate code for system prototyping.

The intent of the next section is not to provide an in-depth introduction to 001 but rather to introduce the components of the 001tool set and to explain how they are employed to support the FODA method. The in-depth languages and syntax are documented in the 001 Tool Suite System Reference Manual [OO1SRM].

This section concludes with a brief discussion of the other tools used to support this study.

3.4.1. Hamilton Technologies, Inc. (HTI) 001

001 has been designed, developed, and used for the rapid development of systems. The 001 technology embodies many aspects of a "Development Before The Fact Approach" [HAMIL86] which assures that a system is developed with built-in quality and productivity. The 001 tech-

nology is based on a set of axioms [HAMIL86] that guarantee consistency and logical completeness of the resulting system design.

3.4.1.1. The 001 Tool Suite

The 001 Tool Suite is an integrated family of automated software tools designed to improve the system development process. The tool suite automates the application of the 001 philosophy to fully integrate data structures, object design, and functional performance.

The 001 Tool Suite [OO1SRM] consists of the following components:

- The 001 AXES language
- A textual editor
- A graphics Map Editor (MapE)
- An Analyzer
- A Resource Allocation Tool (RAT)
  - A function-oriented RAT
  - A type-oriented RAT
- A Systems Management Interface (FACE).
- An Object Map editor (OMap)

The 001 AXES language provides a means of integrating cross-checking and consistency with reuse of data and features. The language includes an object type decomposition via a Type Map (TMap) and functional decomposition via a set of Functional Maps (FMaps). The TMap defines the possible objects and states that an object may have and FMaps are used to define, integrate, and control the transformation of objects from one state to another state.

Both the graphical and textual editors are used to construct an 001 AXES definition. The graphical map editor is used to graphically construct and edit FMaps, TMaps, and the RoadMap (the hierarchy which provides an index to the system of FMaps and TMaps). The textual editor is used to create textual definitions of FMaps or TMaps, or to access the textual version of the graphical definitions maintained by MapE.

The graphical TMap consists of a set of trees. Each tree represents the decomposition of an object. The syntax of a TMap provides four abstract types to represent the decomposition of an object into its component objects. These are the *TupleOf*, *OneOf*, *OsetOf*, and *TreeOf* abstract types.

- The *TupleOf* abstract type identifies an object as consisting of one to a specified number of component objects.
- The *OneOf* abstract type identifies an object as being one of its component objects (i.e.,the object instance may be represented by one and only one of its components objects).

- The *OsetOf* abstract type represents an object as being an ordered set of its component objects. This is similar to the construct of a circular doubly linked list available in some programming languages.
- The *TreeOf* abstract type is used to represent an arbitrary tree structure with an object at each node.

In the graphical FMaps, the 001 user specifies a particular functionality as a tree of functions, with each function specifying its inputs and outputs. In addition, with each function there is an associated control structure specifying constraints on the way that data (inputs and outputs) may flow between the functions that make up that functions' decomposition.

The Analyzer performs the syntax and semantic analysis on partial or completed definitions produced by either the textual editor or MapE. The Analyzer checks to ensure that all parts of the definition are internally consistent and checks all interfaces for correctness and completeness.

After the Analyzer completes the syntax and semantic analysis, the RAT generates operational code.

- The functional RAT generates a target language source code program from successfully analyzed definitions. It insures that the implementation maintains the integrity of its 001AXES definition. It eliminates error-prone hand-coding, permits simulation, and makes rapid prototyping possible.
- The type RAT generates abstract type templates used by the functional RAT. These type templates define the allowed primitive operations on each type.

The System Management Interface is designed to allow easy access to all of the capabilities of 001 and a wide variety of general purpose commands to be executed.

An OMap can be thought of as the runtime instance of data that has been created and organized according to the constraints provided by a particular TMap. The OMap editor is a tool that allows the user to readily access such data and manipulate it in a variety of powerful ways. For example, the Omap editor can be used as a default-form user interface for the system during execution.

3.4.1.2. Use of 001

The analysis of the Movement Control Domain used TMaps to model both the entity-relationships and features of the FODA method. The relationships between entities on the entity-relationship diagram are transformed into the relationships between object entities on a TMap, as seen in Figure 4-4, by representing the "is-a" and "consists-of" relationships with the TupleOf abstract type. The "has" relationship is represented with the OsetOf abstract type.

The feature diagram is transformed into the TMap representation, seen in Figures 4-4 through 4-7, by modeling optional features as leaf node objects of type boolean (or literal), the alternatives as a OneOf abstract type, and mandatory as a TupleOf abstract type. Since a TMap follows the same tree structure as a feature diagram, the concept of "reachability" defined in the FODA feasibility study is maintained within a TMap.

The Fmaps in 001 AXES are used to define the behavior of the system in terms of functional decomposition, control structures, and the flow of data. This information supplements the control and activity information previously represented by STATEMATE statecharts and activity-charts in the FODA feasibility study.

3.4.1.3. Analysis of 001

The main objective of using 001 was to determine the feasibility of incorporating all of the models of FODA into a single tool. The model may then be executed to demonstrate the effectiveness of the method and the ability to specify a system from the model.

In 001, it was possible to convolve the features, entity-relationship, and functional models together with FMaps and TMaps. The combining of Tmaps and Fmaps allows:

- Visualization of the FODA models.
- Consistency checking across the models.
- Implementation of a prototype system, parameterized by features.

Once these FMaps and Tmaps were generated and successfully analyzed for syntax and semantic errors, the m[illegible] was turned into code by the RAT. The type RAT generated a system of object type templates for the domain from the TMap and the functional RAT generated the source code from the integration of the type templates and the FMaps. The code was compiled and executed for prototyping models of a movement control system. This use of 001 is explained in Chapter 5.

The 001 Tool Suite supported the FODA method by enabling a system to be created based on a set of commonalities among similar systems (the FODA model) and the selection of a set of features to be incorporated into the newly defined system.This application of FODA creates an executable prototype of the system with the desired set of features. The user of the prototype may change the performance of the prototype through a selection of different features.

The examination of 001 will continue by looking at the the kinds of user interfaces that 001 can be used with. Currently, a model is executed by selecting the features desired via the OMap editor during execution. However, primitives within the 001 AXES Language enable the model developer to access a graphical user interface (such as Motif) within 001 for the selection of features. This graphical interface would create an environment around 001 where the user could select features and enter object data for model execution without an in-depth knowledge of the 001 Tool Suite or the OMap editor.

During this analysis of the movement control domain, the FODA method still relied on STATEMATE to provide the high-level perspective of functionality and behavior. Further exploration of 001 will include studying 001 Fmaps and RoadMaps as an expression form to be used as an alternative to STATEMATE when expressing high-level perspectives of functionality and behavior. Given our work with 001 to date, we see that although the RoadMap provides some of the high-level functional understanding of the system being modeled, a user of the model cannot easily understand the behavior of the system because of the semantics and

syntax of FMaps. Having expressions of the functional and behavioral perspectives that are understandable in 001 is desirable, as the result would be unambiguous protrayal of intent and full integration with the entity-relationship and feature definition side of the problem currently expressed in Tmaps.

3.4.2. Use of STATEMATE for Movement Control Analysis

The use of STATEMATE in this analysis was the same as that described in the FODA feasibility study. STATEMATE Statecharts and Activitycharts were used to represent the high-level functional model. Changes from STATEMATE Version 2.5 to Version 3.0 are reflected in the charts. In STATEMATE Version 3.0, the issues of decluttering the charts and the visibility or scope of the elements within a chart are addressed. These changes and their impact on the process of modeling a system are documented in a separate report [SEI91b].

3.4.3. Info Mode Under the GNU Emacs Editor

The movement control domain analysis used *Info*, a documentation tool, to capture much of the information about a domain: the features descriptions, the entities and relationships descriptions, and the terminology dictionary. These pieces of information are related to one another in ways that paper documents cannot easily capture. Thus, the creation of a file of the domain information formatted for use in Info mode under GNU Emacs [STALL87] allows for the navigation through these collections of textual information in a useful way.

GNU is the name of a set of (virtually) free software packages available for use on computers that run the UNIX operating system. GNU is *not* public domain software, but its copyright permits its users to modify and/or redistribute it almost without restriction. GNU Emacs is a powerful editor that has the ability to support various modes of operations:

- Normal text editing and use of various document description language (T_EX and Nroff for example).
- Program code development and formatting (C and Lisp are supported).
- Support for various utilities available for use within the editor (Mail and Shell).

Info is one of the utility modes. It provides a powerful facility for navigating among many pieces of text that are related to one another in some way(s), something like a primitive form of hypertext. The *Help* facility for Emacs is written for use in this mode.

Appendix C describes the basic information on how to format a file for use within Info mode and a summary of how to navigate in an Info file.

4. Movement Control Domain Analysis Review

This chapter presents the products of the context analysis and domain modeling phases of the movement control domain analysis. The product presentations are intended to give the reader an overview of:

- The style and content of the products.
- The knowledge of the domain represented by the products.

Many of the products are too large to be discussed in detail in the body of this report. In each case, the section describing that product will refer the reader to the appropriate appendix for more information on that product. This is to allow the reader to get a high-level understanding of the domain as portrayed in the products.

4.1. Refinement of the Movement Control Domain Scope

The purpose of the context analysis phase of domain analysis is to establish the scope of the proposed domain. In order to understand the scope of a domain, we must first have a basic knowledge about the higher level domain(s) of which the target domain is a part. Also, an understanding of the domain's users and related applications is useful in knowing how the target domain interfaces with its peers.

This Context Analysis report [SEI91a] details the results of the context analysis for movement control. The domain modeling phase has addressed most of the issues raised in Section 3.2 of that report. Several of the issues that produced notable changes to the perceived context for movement control, or changed the context analysis products as they were previously published, are summarized below:

- The usage of movement control software extends beyond the movement of units in the battlefield. Such software is also needed for *logistical* movement, which is the movement of supplies and equipment to support and sustain the units in combat. The considerations of logistical movement require the ability to view the assets available for performing that movement. Organic assets (those intrinsic to the unit) are used with unit movement. Logistical movement requires knowledge about common user assets so that an appropriate selection can be made. Most of the functionality is equivalent to that needed for unit movement, so that inclusion of logistical movement does not greatly expand the functionality to be incorporated into the domain model. Even at the level of strategic movement where large groups of men and materiel of all types are moved over great distances, the basic features and entities are applicable.
- A large amount of the movement conducted on the battlefield is not planned and scheduled in the time frame of generation and promulgation of an Operational Plan or Order (OPLAN or OPORD). Section 3.2.1.d of the Context Analysis report discusses the issues of timing. Most of the movement that occurs on the battlefield is in response to commanders

receiving new missions from their superiors.

Based upon the mission, the commander perceives a need to move to a new position and informs his superior of this intent via a movement request. The higher echelon staff approves the movement after assuring that it does not conflict with other current or planned movements or actions. The planning for a unit's movement is normally done by the unit, not some staff person higher up the chain of command.

- The *context* features affect many of the operations features, more so than was reported in the Context Analysis report. Section 4.3.2.2 describes the coupling of these two sets of features.
- Movement feasibility or estimation of movement capability is an issue that was only dealt with at a cursory level during the domain modeling phase. Although this issue is important at the higher echelons of command for strategic planning, the project consciously decided to focus its effort on movement control within ATCCS, which is a tactical movement system. In such systems, feasibility is implicitly determined by the unit actually planning its move. The domain model contains features that incorporate the concept of movement feasibility, but this feature is not analyzed to the same extent as other areas within the domain.

The next section will establish the placement of movement control software within the software structure of ATCCS and its constituent nodes and show the refined context diagram for the movement control domain.

4.2. Context Analysis Products

Movement Control is a software application common to the ATCCS BFAs. The products of context analysis place the movement control domain within the overall ATCCS structure and establish a common pattern for movement control operations.

4.2.1. The ATCCS Development Strategy

The ATCCS software development is implemented using a hierarchy of layers as seen in Figure 4-1. The figure is in the form of a structure diagram that is described in Section 2.1. The four primary layers of ATCCS are the hardware layer, Common Off-The-Shelf (COTS) layer, Common ATCCS Support Software (CASS) layer, and the Command and Control Applications layer.

Layers 1 and 2 of the figure show the Common Hardware and Software (CHS) suites which provide a powerful operating environment for each computer that is part of ATCCS. The two layers provide low-level interoperability to allow networking of the computers running the BFA software on each node. Layer 1, the hardware layer, consists of common hardware components. Components within this layer include central processing units, printer devices, communication devices, and display devices.

Layer 2, the COTS layer, consists of commercial software, already built and tested, which can be tailored to provide software services to the BFAs. Examples within this layer are operating system (UNIX), database management software, graphics support, communication drivers, etc.

The line between Layer 2 and Layer 3 symbolizes the standard interfaces to packages such as the UNIX operating system, the Structured Query Language (SQL) for relational databases, and graphics packages like the X Windows System. Packages such as these provide a high degree of software portability as the hardware platforms evolve over time.

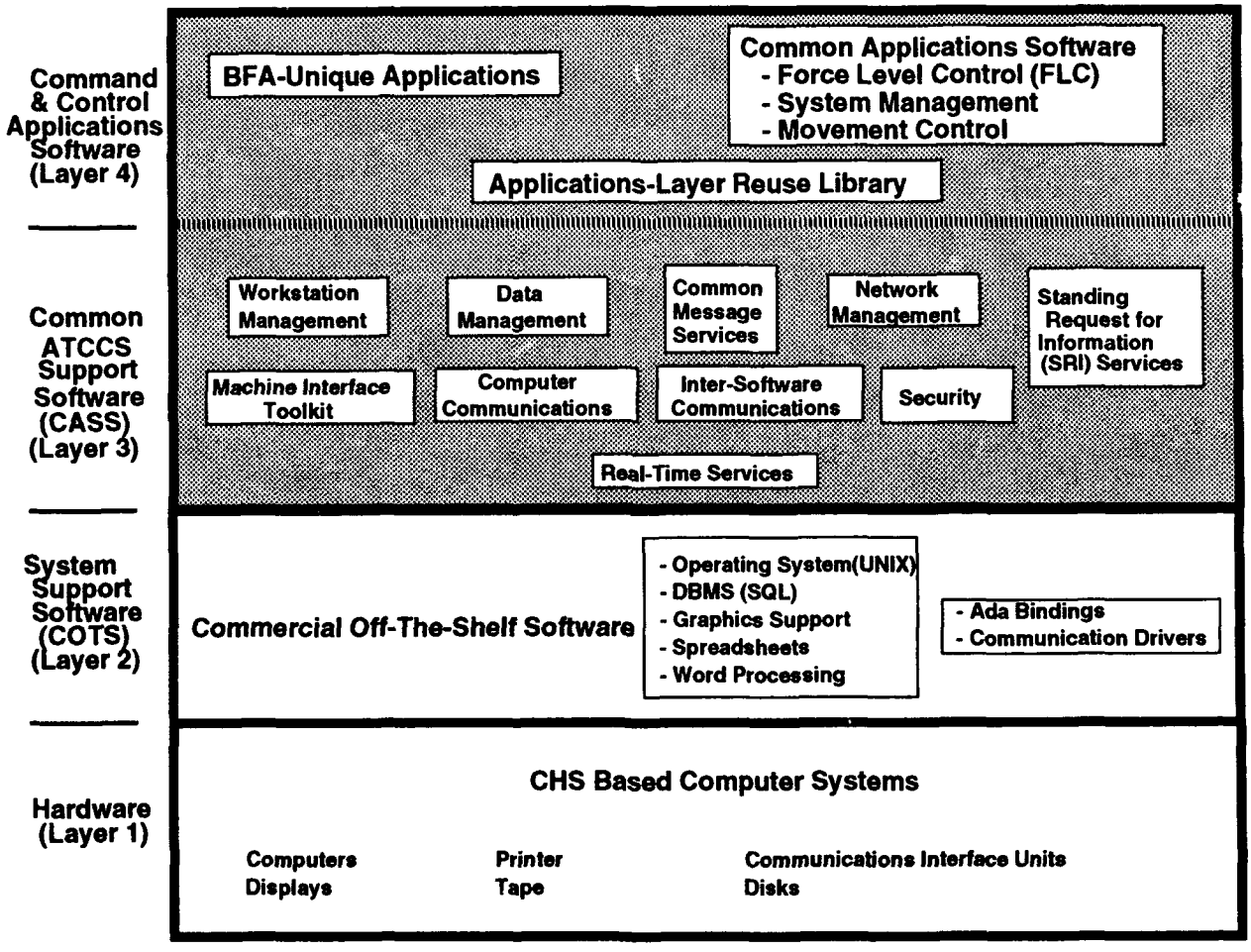


Figure 4-1 The Movement Control Structure Diagram

Layer 3 shows the CASS, which provides many common services such as message handling and system and data managers. This layer provides for higher maintainability and extensibility of the ATCCS software. The CASS layer provides common interfaces to the application software in the applications layer from the COTS and hardware services. The major thrust of CASS is to "shield" the application software from changes within the COTS or hardware layers. The CASS layer consists of several classifications of services, database management services, display services, communication services, operating system services, etc. Each of the ser-

vices within the CASS layer contains various objects. These objects provide the lower level computational routines and data structures needed to support the application software. The dotted line between Layers 3 and 4 denotes that the CASS software is an integral part of the ATCCS systems, as opposed to the COTS applications in Layer 2 which are not developed and maintained by ATCCS.

Layer 4, the Applications layer, consists of software that provides common C^3I functionality, as well as unique services to the five BFA systems being developed (AFATDS, ASAS, CSSCS, FAAD, and MCS). Although each of these five systems are targeted for a specific functional area, they share common application requirements, some of which are Target Damage Assessment, Resupply Operations, Order of Battle, and Movement Control. The recognition of these common applications within the ATCCS systems prompted the designers of the ATCCS architecture to add a common application section within this layer.

The layered architecture described above is one where the bottom layers are at low levels of abstraction and the upper layers depend upon the use of appropriate interfaces into the lower level to achieve their functionality. Such architectures are becoming common in building software. The OSI communications architecture [YOUNG89] and the X Window System [ZIMM80] are highly visible examples. Layered architectures are one way of removing extraneous (outside of the domain's scope) functionality from consideration during the domain analysis so that the focus is on the specific needs of the selected domain.

4.2.2. Context Diagram

Movement control is concerned with the movement of battlefield assets in support of tactical objectives. As noted in Section 2.3.1 of the Context Analysis report [SEI91a], movement control plays a role in each of the three key operations (plan, direct, execute) within a C^2 system, and these key operations carry over into the context diagram shown in Figure 4-2.

- Movement control receives inputs as *commander's guidance* during the **Planning** function, checks the requests against the *constraints and options* data, and generates the *movement estimate.*
- Alternately, a unit may receive a *movement requirement* due to operational considerations. The unit plans it's movement resulting in a *movement request.*
- The movement request is matched against actions occurring on the battlefield by the area commander's staff, using information available from the *operational considerations* flow.
- Finally, movement control generates *orders* using the **Direct** function as output to the **Execute** function. The orders also are *stored* so that their *status* (for example, in progress, completed, or canceled) can be requested or updated via the *movement status* inputs from **Position Reports.**

The computation of movement feasibility (the resources needed, routes to be followed, etc.) requires that two types of information be available:

1. *Static* data: Information about the terrain over which movement is to occur (roads, mountains, rivers, bridges, etc.), measures of unit sizes and payload capacities, and information about tactical considerations that limit or constrain movement options.
2. *Dynamic* data: Current information about the weather, terrain, or road conditions (the effects of weather or combat actions), the availability of transportation assets, and data on friendly and enemy units (locations, sizes, etc.).

Most of this data is produced or maintained by operations outside the scope of the movement control domain. For example, all data about weather, terrain, and the disposition of friendly and enemy forces is obtained directly from intelligence.

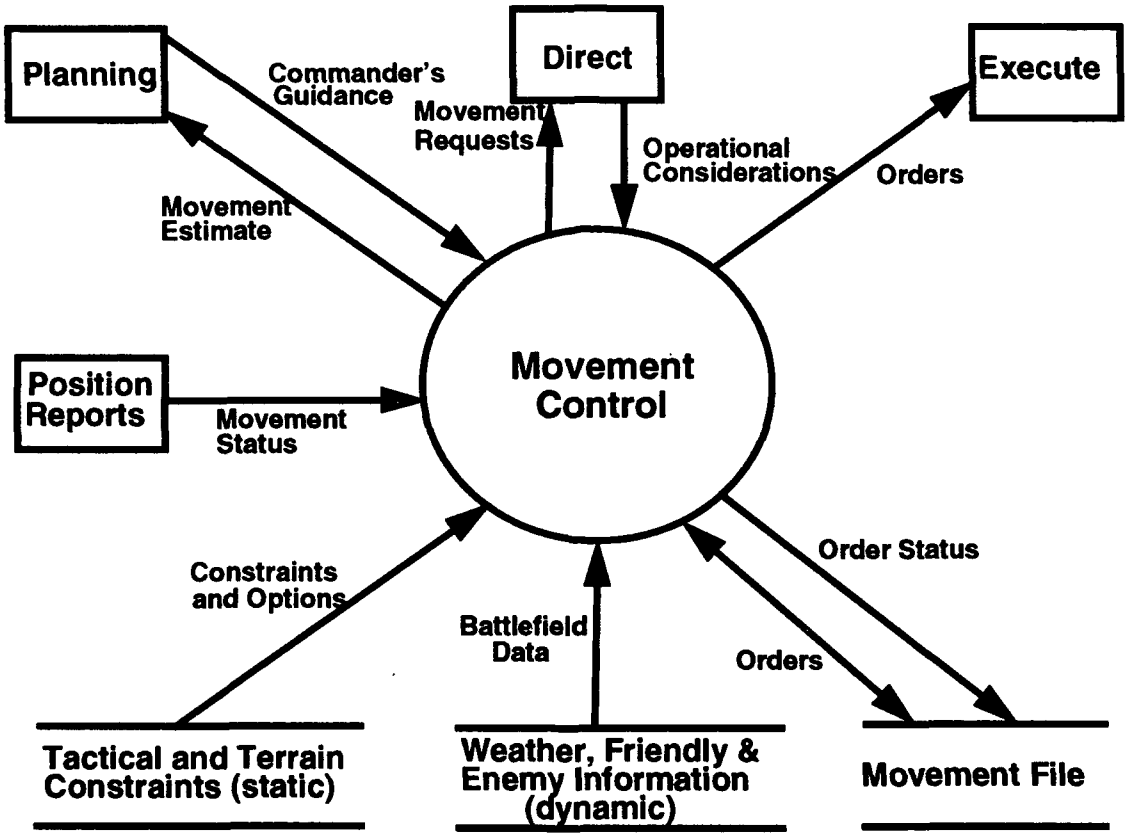


Figure 4-2 The Movement Control Context Diagram

4.3. Domain Modeling Products

Two important results of the domain modeling process are the modeling of the entities and features of the domain. The next two sections present the highlights of these models using the 001TMap notation as described previously in Section 3.4.1 of this report[1]. The third section describes the functional model using STATEMATE as described in Section 3.5.2. Lastly, the

[1] Due to limitations within 001, the entity or feature names appearing on the TMap may be shortened versions of the equivalent name appearing in this chapter for discussion purposes.

domain dictionary is described. Each of the products is fully detailed in one or more appendices as indicated in Table 5.

Model	Reference
Entity Relationship data	Appendix - D
Feature data	Appendix - E
Feature Catalog	Appendix - F
Functional Model	Section 4.3.3.
Domain Dictionary	Appendix - G
Domain Acronyms	Appendix - H

Table 5 List of Domain Model Products and Report Appendices

4.3.1. Entity Relationship Model

The ER model, depicted in Figure 4-3, presents system entities and relationships between them. The information contained in the ER model is largely derived from Army doctrine[FM 101-5]. The road movement table, defined by the field manual, pertains to commands issued by the commander to units regarding movement between specified locations. The commands contain an enumeration of vehicles, organized as a convoy, the routes the convoy is to follow, and the schedule for carrying out the movement.

The figure uses the entity relationship notation seen in [SEI90a] and shows that the majority of relationships are associated with three kinds of entities:

1. *Commander.* The person making decisions about how missions are to be accomplished.
2. *Units.* The combat or support organizations a commander has available for accomplishing missions.
3. *Orders.* The mechanism by which the commander communicates his decisions to units for implementation. A particularly important kind of order is the *Movement* (Mvmt) *Order.* Generating a movement order requires dealing with *Transportation*, *Routes*, and *Schedules* at various points in the process.

This high-level representation of the ER model is sufficient for an overview of the entities in the domain. An ER modeling tool would be required in order to manipulate detailed ER data, maintaining completeness and consistency. No modeling tools that support the FODA approach to ER modeling, which combines ER modeling with semantic data modeling, were found at the time of the domain analysis (see Section 5.2 of [SEI90a] for further information). The 001 tool does, however, provide support for semantic data models in its Tmaps, which

constitute the bulk of the information that needs to be captured in the FODA ER model. The 001 tool still lacks a means for depicting information on relationships between entities in the semantic network. The 001 graphic notation is therefore supplemented with textual information as detailed in Appendix D.

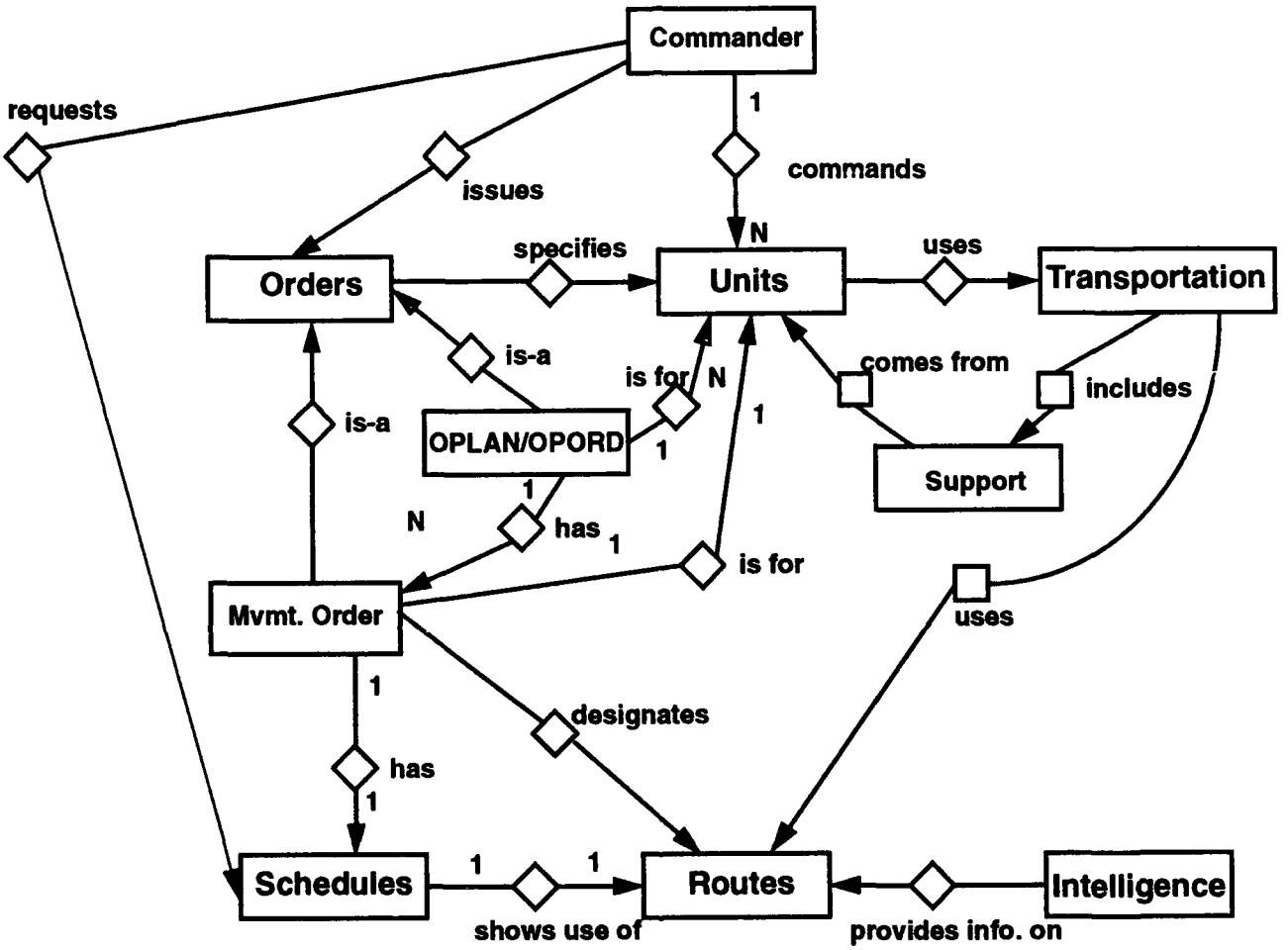


Figure 4-3 Entity Relationship Model

The top level of information found in the movement control ER model is portrayed in Figure 4-4 using the 001 syntax. The figure shows that the movement control domain consists of the six following types of information:

1. *Orders.* The entities that are sent and received by various command echelons to communicate information about the movement control process (e.g., Operational Orders).
2. *Transportation.* The entities that represent the vehicles that will carry the unit or material from its current to its future location.
3. *Schedules.* The entities that provide a time base coordination of present and future use of transportation infrastructure.

4. *Distribution Plans.* The entities that represent the plans of the commander and his movement control staff to control the use of important transportation infrastructure to best accomplish the overall mission.

5. *Intelligence.* The entities that represent the knowledge of the battlefield state; most importantly, the information about the transportation infrastructure and status.

6. *Task Force.* The entities that represent the units or sets of units needing to move or requiring the movement of various materials.

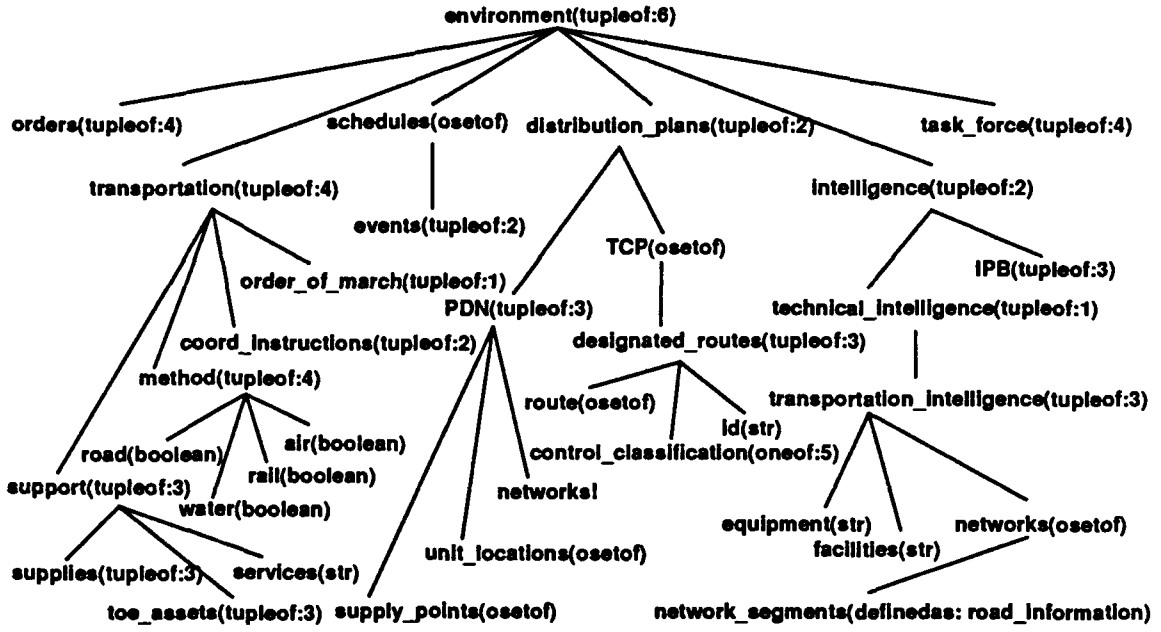


Figure 4-4 001 Representation of Top-Level Entities

Important aspects of the movement control problem are captured in the *Transportation*, *Distribution Plans*, and *Intelligence* entities. *Intelligence* comes in two forms that are relevant to movement control: technical intelligence and Intelligence Preparation of the Battlefield (IPB). There are several categories of technical intelligence; however, the most important to the movement control domain is transportation intelligence, which is information about the equipment, facilities, and networks available for use in meeting transportation needs in a theater of operations. The road network is particularly important as the vast majority of movement within a theater is performed on roads. IPB is "a systematic and continuous process of analyzing the enemy, weather, and terrain in a specific geographical area."[FM 34-1] IPB has three parts of its process that provide information relevant to movement control:

1. *Terrain analysis.* Gathers and analyzes information pertinent to how the geography of the theater area will affect operations. For movement control, this involves the checking and updating of terrain (including roads, bridges, tunnels, etc.) data collected by the Defense Mapping Agency.

2. *Weather analysis.* Plays an important role in understanding the usability of parts of the road network and thus, the capability of the network to sustain planned movements.

3. *Threat evaluation.* Seeks to predict the actions of the enemy forces and their consequences. For movement control, this may mean the rerouting of convoys around areas where fighting may occur as well as moving friendly units in those areas.

Distribution Plans are the entities that define roads and other aspects of the transportation infrastructure (e.g., rail lines, supply points, etc.) movement control within the theater area. The *Physical Distribution Network* (PDN) is used primarily by logistics personnel to visualize where the combat units are, where the supplies they will need are, and the paths between the two. The *Traffic Control Plan* (TCP) is constructed by the command responsible for movement control in a particular area. It designates specific routes, most of which are expected to carry a high traffic load, and places controls and monitors on those routes to enforce compliance.

Transportation is organized information about a movement, consisting of four entities:

1. *Support.* The need for other units to provide supplies (food, water, fuel, etc.), TOE assets (assignable equipment or personnel of any kind), or other services for a particular movement.

2. *Method.* Selecting the network and associated equipment for carrying out a movement. During the analysis the focus was on road movement, although most of the information (and most of the feature model) is applicable to the other three modes of transportation. At the theater level, a significant portion of planning distribution patterns involves mode determinations of various unit and logistical movements.

3. *Coordinating Instructions.* Information about aspects of a movement that the moving entity needs to know about measures taken to help assure the safety of the movement.

4. *Order of March.* Information about the characteristics and organization of the vehicles in a convoy to be used in performing a movement.

Each of the three entities: Intelligence, Distribution Plans, and Transportation, describe the data used as inputs to the process of planning individual movements. Intelligence provides the movement planner with data on potential paths (segments and terrain), the Distribution Plans augment this path information with data on controlling the usage of important road segments, and Transportation provides information on the potential kinds of Support that may be requested. Transportation also provides entities for storing some of the important products of the movement planning process, the Method selected, the Coordinating Instructions needed, and the Order of March to be followed.

Appendix D gives the complete graphical depiction of the ER model as seen in the 001 Tmap and gives a textual description of each entity given in the ER model.

4.3.2. Feature Model

The window manager feature model was portrayed in the FODA report using both a textual description and graphical representation in the form of an And/Or diagram. The textual description used in this report follows that of the FODA report. The format is only slightly changed for inclusion into a file for use in the GNU Emacs Info mode (see Appendix C for a discussion of this tool). The graphical notation was modified to incorporate the 001 T-map notation. A description of the use of the 001 notation for describing the basic relationships between features (mandatory, alternative, and optional) is given below.

Feature Type	001 Notation in Graphical Representation
Mandatory	The *feature_name*(**tupleof:X**) notation is used for parent features where X is the number of child features; feature_name' signifies a leaf feature.
Alternative	The *feature_name*(**oneof: X**) notation is used for parent features where X is the number of alternate children.
Optional	Optional leaf features are denoted by the **(boolean)** notation after the name.

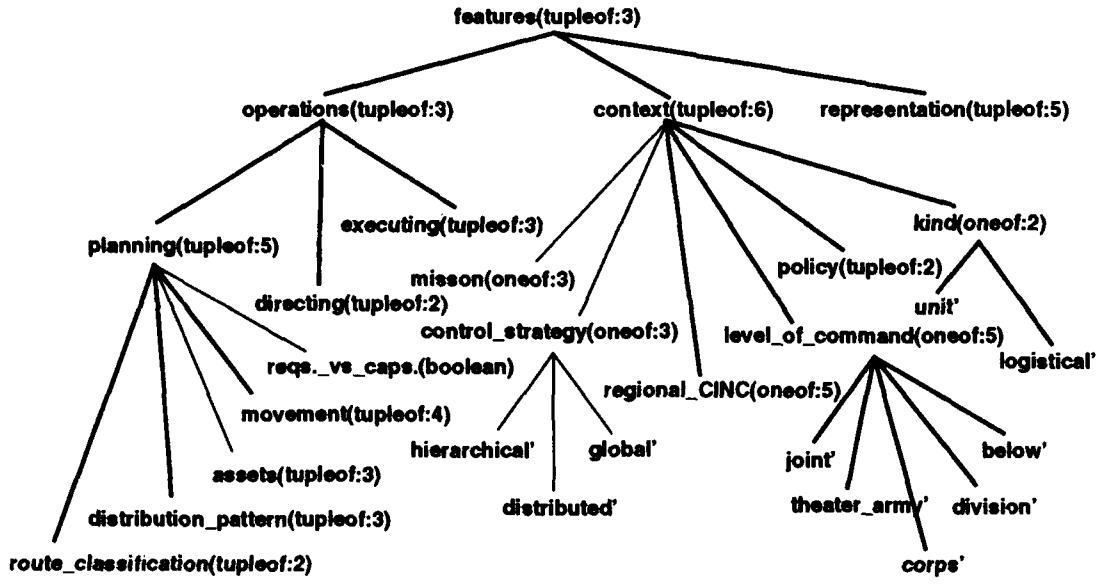


Figure 4-5 The Top-Level Feature Model

As shown in Figure 4-5, there are three main groups of features within the movement control domain:

1. *Operations*. Those features that describe the functional characteristics of movement; the services that a system must provide.
2. *Context*. Those features that describe the overall mission or usage patterns of a system; the description of the class(es) of users for a system.
3. *Representation*. Those features that describe how information is viewed by the user or produced for another system; what sorts of input and output capabilities are available.

Each of these groups of features is described in more detail in the following paragraphs.

4.3.2.1. Operational Features

The operational features fall in three major categories:

1. *Planning*. Those features that describe the process of preparing for future assigned or assumed movements.
2. *Directing*. Those features that describe the operations needed to transition the results of Planning into forms of communication that convey information governing actions, i.e., orders.
3. *Executing*. Those features that describe operations that take place when a movement is occurring.

The Planning features are the most important as they denote the features that are seen as the essential elements of movement control. The Planning feature is decomposed into five sub-features:

1. *Route Classification*. Those features that allow for determining the suitability and capacity of a road or set of roads for use by various kind of vehicles.
2. *Distribution Pattern*. Those features that characterize the ability to view movement requirements at an aggregate level and allocate road or other means of transportation to the continued coverage of those requirements. Much of this problem involves the ongoing relocation of materiel from its arrival into the theater area through supply points to units using the materiel.
3. *Assets*. Those features that allow movement planners to consider different usage characteristics for the vehicles, networks, and equipment to be employed in movement.
4. *Movement*. Those features that characterize the functionality of the most important and frequently performed processes in movement control, the planning of individual moves. The decomposition of these features is depicted in Figure 4-6 below.
5. *Balance Capabilities versus Requirements*. A feature that allows command personnel the ability to determine whether or not the transportation system can support the requirements of a plan.

The features under movement describe the essence categories of movement functionality. Highway_regulation is shown as a single feature broken in two sub-features, traffic_control and deconfliction. Traffic_control is the physical regulation and control of designated routes, performed by military police or civil authorities. Deconfliction is the coordination of all movement within a specified network to ensure that no two sets of vehicles will occupy the same space at the same time. Deconfliction is an important part of a comprehensive movement control solution because it is necessary to keep moves from interfering with one another and to track the utilization of key routes within the network.

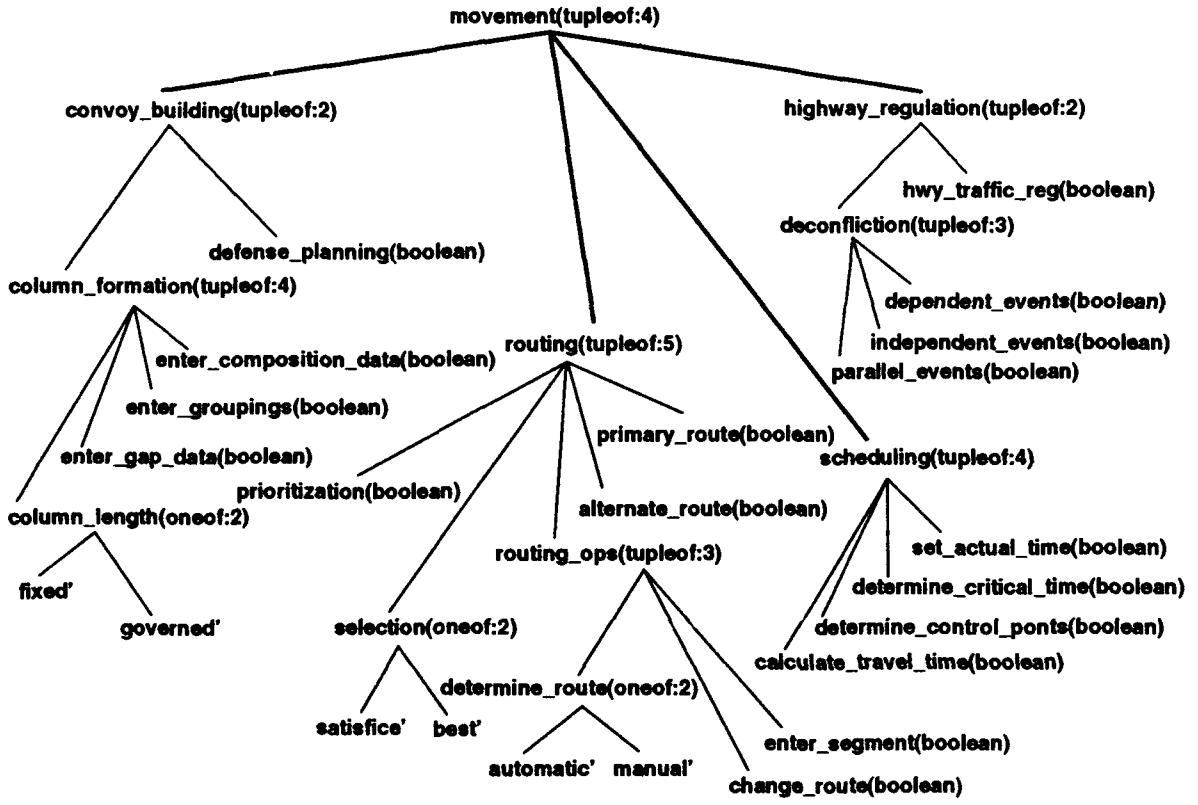


Figure 4-6 The Detailed Operational Features of Movement

The process of planning a convoy requires three operational_features: convoy_building, routing, and scheduling. Convoy_building has two sub-features, defense_planning and column_formation. The column_formation feature, a major part of the convoy planning process, supports convoy_building through the four following features:

1. Enter_Composition_Data. This is data about the number and types of vehicles to be made part of the convoy being formulated.

2. Enter_Groupings. This is data about the placement of the vehicles and various groups of vehicles within the convoy. Army movement doctrine requires that specific rules must be followed in the organization of vehicles into convoys. [FM 101-5] provides a detailed description of this doctrine.

3. Enter_Gap_Data. This is data about spacings between vehicles and groups in the convoy for safety, security, and other considerations.

4. Column_Length. After data about the vehicles, groupings, and gaps is obtained, length is determined so that the amount of road space needed by the convoy is known. This information is used by scheduling.

Routing is the determination of a viable path from one location to another. Three of the five sub-features under Routing pertain to the flexibility of this feature's implementation: its ability to consider multiple routes and to use the priority system to enable or eliminate the use of various segments that are restricted by other movement control organizations. The two features of interest are the Determine_Route feature under Routing_Ops and the Selection feature.

1. Determine_Route. Most systems still require the manual entry of routing information, even if this is just pointing and clicking on path segments on a display map. However, some systems allow for automatic routing where the user need only specify the start and end (release) points desired and a usable path between those points will be generated.

2. Selection. The selection feature is a control over the amount of processing that the system may perform in path selection. In satisfice selection, the computations stop when one viable path has been encountered or verified. In best selection, the system will perform a more exhaustive search, seeking the optimal path based upon user defined characteristics such as path length or road type.

Scheduling is the determination of the time span during which the movement will occur. This feature includes operations to set the times the convoy will occupy various parts of the road space as identified by the routing operations. The scheduling feature will include operations and information that are well understood from the scheduling domain. This information correlates to the four sub-features:

1. Set_Actual_Time. This feature allows for scheduling a series of events to occur in a specified time sequencing in relative time and later, for setting the true start time for the sequence and compute the rest of absolute or real times.

2. Determine_Critical_Time. This feature allows for the designation of the single time from which other times in the schedule are based, either the time of the first or last event.

3. Determine_Control_Points. This feature allow for entry of time delays for stops at various point along the route for resting, re-fueling, etc. This is time lost in activities that do not contribute to progress in the schedule.

4. Calculate_Travel_Time. Complete the schedule, computing results as needed.

Routing and scheduling are two interrelated domains that have been the subject of much research, including work on solutions to notoriously difficult problems in computer science such as the Traveling Salesman Problem. The difficulties of the movement control routing and scheduling problem stem from the need to be responsive to *dynamic* demands, because many movements are generated as a result of changing battlefield conditions. Yet most routing and scheduling algorithms require a *static* (prearranged) set of movements to be accomplished, and attempt to provide as optimal a set of paths and schedules as possible [BODIN81]. Addition of a new movement requirement may yield a completely different solution. However, one of the unwritten axioms of movement control is to not change the route or schedule of approved movements. The need to cope with continuous requirements for new movements without disrupting the current set of approved movements presents a problem for `tactical` or `operational` movement control that is not solvable using standard techniques. These techniques are applicable to `strategic` movement as the planning time frame allows for a more complete set of movement requirements to be generated.

4.3.2.2. Context Features

The context features, shown in Figure 4-7, provide a means of describing the users of movement control systems and the missions they will perform. During the domain modeling process, these features provide the primary means of understanding what movement operations are needed by various echelons of command and how operations are affected by the overall situation.

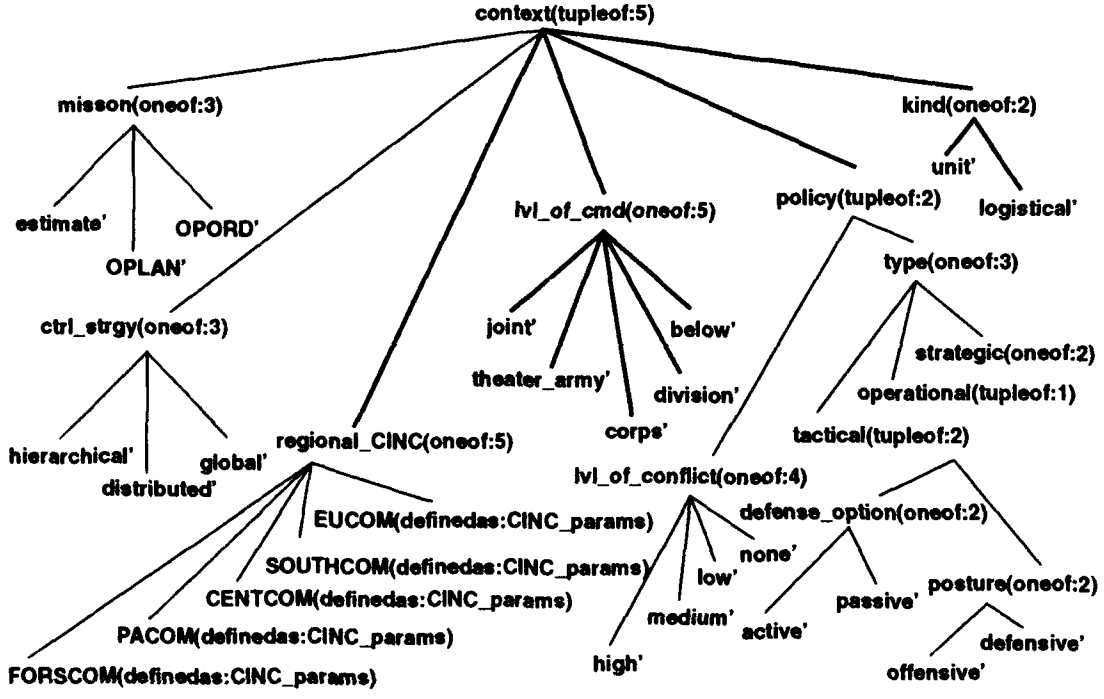


Figure 4-7 The Context Features

The six top-level context features are:

1. The `mission` feature defines the accuracy requirements needed for the forthcoming calculations. Early planning and feasibility computations do not need the completeness and accuracy of data that final orders require.
2. The `control_strategy` feature defines the echelons and systems using movement control software. Selection among the given alternatives affects the resulting software structure. How the structure is affected is explained in more detail later in the description of the functional model, Section 4.3.3.
3. The `regional_CINC` feature defines information and parameters for the domain that are specific to a given region or location, such as the use of metric measurements in Europe versus English measurements in the United States, or differing message formats used for compatibility with host nations or other preexisting systems.
4. The `level_of_command_feature` defines the distinctions in various aspects of movement control that occur at the different echelons of command. The scope and scale of the movement problem is considerably different at joint commands (such as NATO) than at a brigade headquarters coordinating its subordinate units.
5. The `policy` feature defines most of the situational attributes that affect movement control. For `tactical` movement, the `posture` and `defense_option` features define changes in operations that Army doctrine implies should hold in various situations.
6. The `kind` feature provides a mechanism to keep a clear distinction between the differences in `unit` and `logistical` movement as seen during the context analysis phase and its results.

4.3.2.3. Representation Features

The representation features describe the mechanisms available for getting information to and from the user of a system. For movement control, there are five potential ways to represent data as shown in Figure 4-8 below.

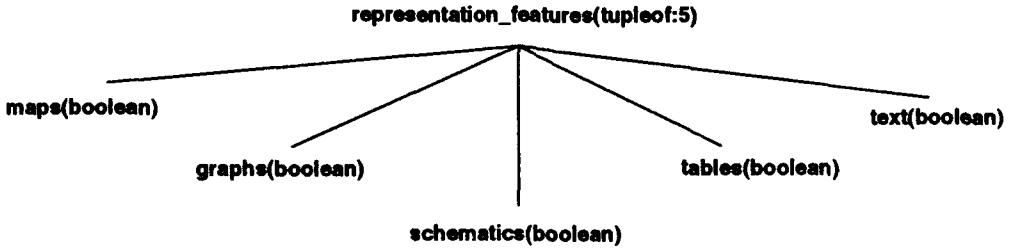


Figure 4-8 The Representation Features

1. `Maps`. This feature requires that several important capabilities be present in the underlying services of the computer systems on which movement control software is to be run:

- A mechanism for storing and retrieving geographic information including location, terrain data, named entities, and much more.
- A graphical display environment powerful enough to provide unique shadings, colorations, outlines, etc., to allow the display of many varied types of data in distinct forms.
- A mathematical package to support the analysis of geographic data.

2. `Graphs`. This feature allows the user to view the timetable for one or more movements using bars to represent the passage of units over portions of a route. This timetable is called a movement graph and is used by movement control personnel to visualize a movement or the usage of a high traffic route by multiple convoys.
3. `Schematics`. Another graphical representation option which allows users to build a complex graph of paths between points and the flow capacities and quantities along those paths. Various schematics are used in the process of distribution planning.
4. `Tables`. An alternative to graphs as a method for viewing a timetable for one or more movements. Similar to Text below, but more powerful because a mechanism for formatting data is implied.
5. `Text`. Any representation (ASCII or other) that allows use of alphanumeric characters and other symbols used in typewritten or intercomputer communications.

Appendix E shows the complete feature model for the movement control domain in the 001 Tmap graphical form and gives a textual description of each feature in the model. Appendix F contains the features catalog that shows what features are used in the systems analyzed during the domain modeling phase.

4.3.3. Functional Model

Movement control is one component of a complete tactical command and control system. These command and control systems are characterized by the need to:

- Monitor the battlefield state.
- Compare that state to a desired state based on long-term goals.
- Direct actions to change state.
- Execute actions that actually change state.

The process runs continuously. The context analysis produced a process control loop to show the relationship among these control steps and system states.

While process control models an ideal view of command and control, the functional model within the FODA method models actual systems and data requirements. The important aspects of the functional model are:

- High-level activities that characterize the domain.
- Control structures that model the behavior of the systems in the domain.
- Information structures that are common to those systems.

The FODA method uses STATEMATE activity and state charts to document the functional model.

The features within the domain define commonality and differences among activities, control, and data. The functional model utilizes these features as the basis for common activities and control and to parameterize control structures which result in different patterns of behavior. This approach allows the functional model to highlight commonality at a higher level and isolate mission details particular to a given system. These details are captured primarily in the context features and the selections among optional/alternative operational features. Mission details determine which operations are ultimately performed in an individual system and the specific data produced or consumed by these operations.

The STATEMATE representation of the functional model provides a form for expressing movement control operations, executive control, and data requirements. This representation is documented in a hierarchy of activity and state charts. Activity charts show the operations within movement control and their information flows, and Statecharts show the behavior of these activities according to particular mission characteristics (context features and optional/alternative operations).

4.3.3.1. Movement Control as Part of Command and Control

The highest level activity chart places movement control within the context of a command and control system. Figure 4-9 shows the interaction of movement control with both Army intelligence and operations. (These activities are labelled movement_control_activities, trans_intel_act, and operations_activities in Figure 4-9.) The relationships among these three activities could be extended to a complete command and control system, including domains parallel to movement control such as force level control or fire support planning. The same interrelationships with intelligence and operations would exist for these other domains.

Movement control must have intelligence and operations information to carry out movement activities. The information flows shown on the activity chart (battlefield_data, operations data, and cmdrs_guidance) describe the data exported by intelligence and operations activities and imported by movement control. These flows are also shown on the context diagram of Figure 4-2. The contents and structure of this information is derived from the entity relationship model. Use of the entity relationship model to structure the data will permit commonality between movement control systems and other related applications. Provided the appropriate information flows exist, the movement control activities are not dependent on any particular implementation of either intelligence or operations. Movement control sees only the data they export and the data movement control returns to them.

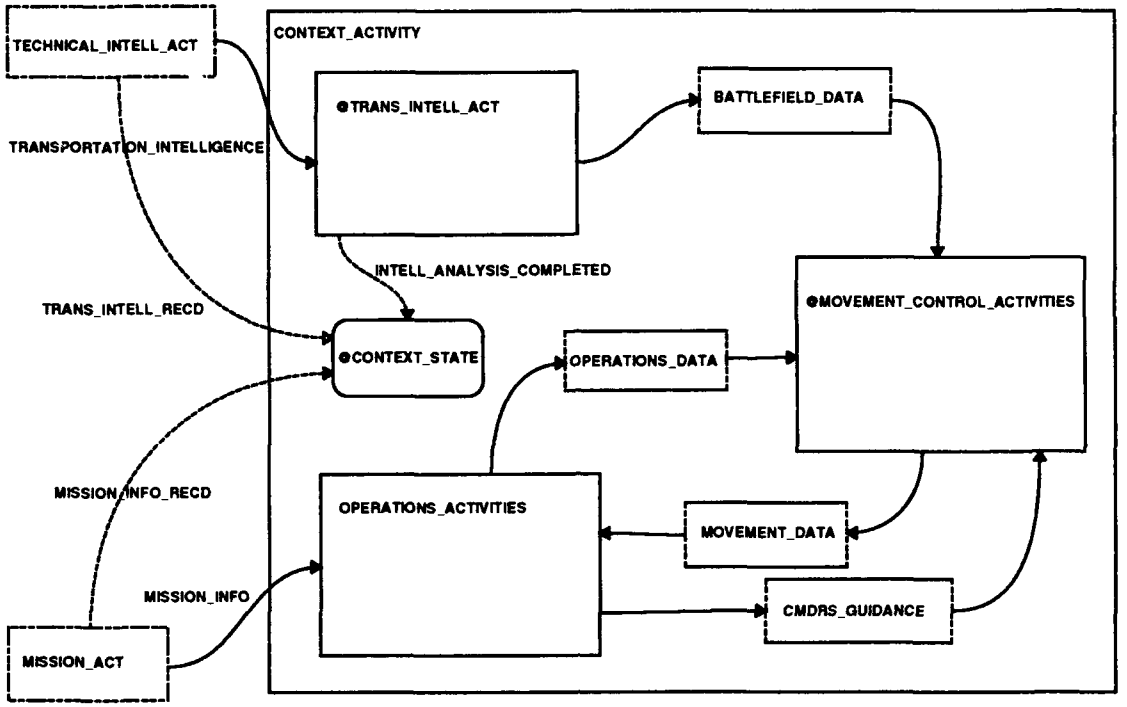


Figure 4-9 Movement Control Activity Chart

The box labeled context_state controls the behavior of intelligence, operations, and movement control activities. Figure 4-10 shows the state chart that represents context_state. The form of this chart indicates that each of the three active states: intelligence_behavior, operations_behavior, and movement_control_behavior are running in parallel. These states, in turn, control the parallel behavior of the intelligence, operations, and movement_control activities in Figure 4-9.

This parallelism supports the common structure within the command and control system. Actual Army staff operations must proceed in parallel with appropriate sharing of information. Automated command and control systems must support this concept of operations. Movement control operations are driven by specific staff functions (e.g., the issuing of an order or the receipt of new intelligence information). The chart shows the initiation of movement_control_behavior in response to a change in battlefield state indicated by integrate_intell or mission_order_issued. In fact, this parallel structure also applies to other subsystems within a command and control system. Mission orders and intelligence will affect their operations, as well.

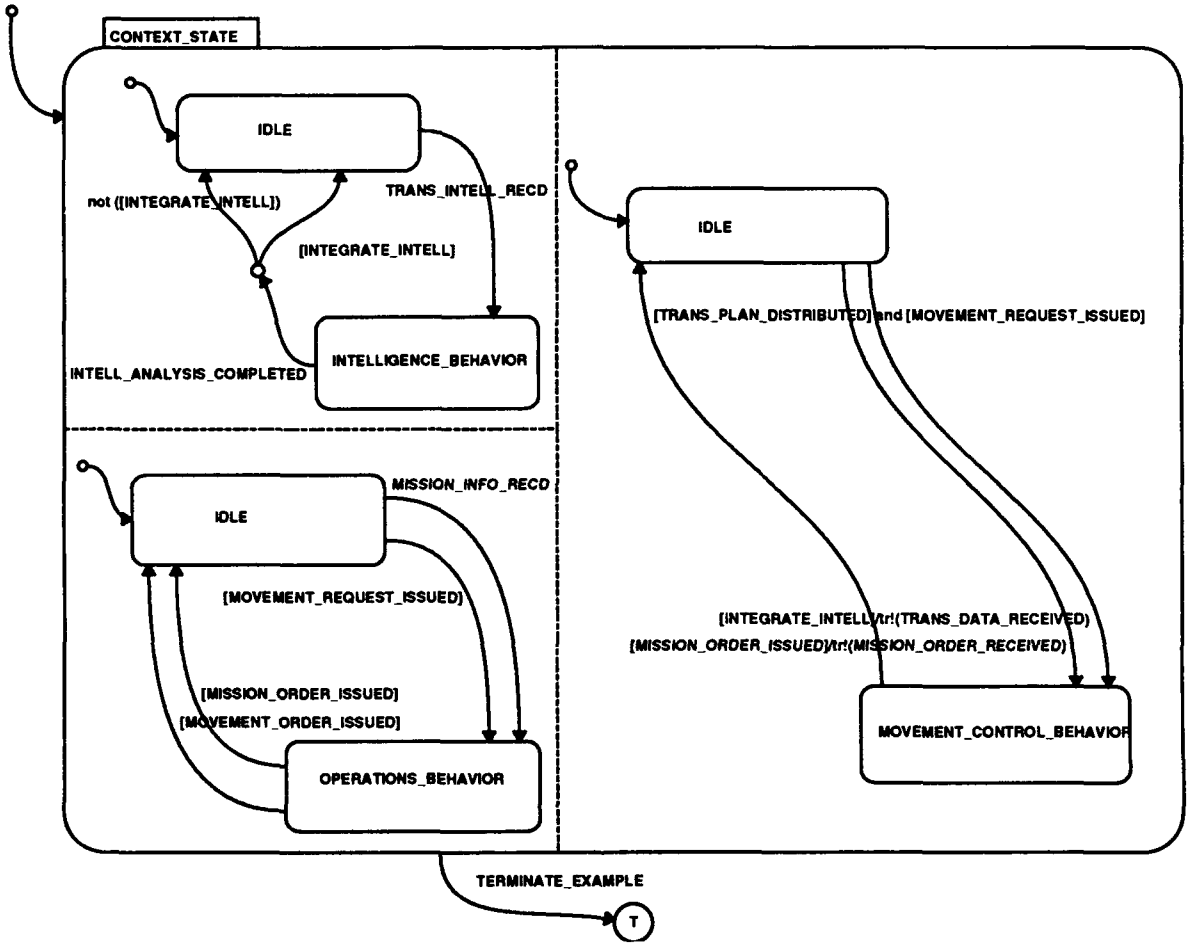


Figure 4-10 Behavior of a Command and Control System

4.3.3.2. Common Intelligence and Movement Control Activities

Each of the activities within Figure 4-9 may be decomposed into lower level activities. For example, the decomposition of intelligence activities and behavior is based on Army doctrine [FM 34-3]. While the detailed operations within each activity of this chart will vary depending on the specific intelligence system, the pattern of activities and their interactions will be repeated from system to system. This common activity structure is illustrated in Figure 4-11.

The domain analysis also revealed the common structure for movement control activities shown in Figure 4-12. These activities are derived from the operational features and the study of commonality among related movement control systems. The activities of this chart perform operations defined by specific features:

- Highway_regulating implements `highway_regulation` operations.
- Deconflicting_route implements the `deconfliction` feature.

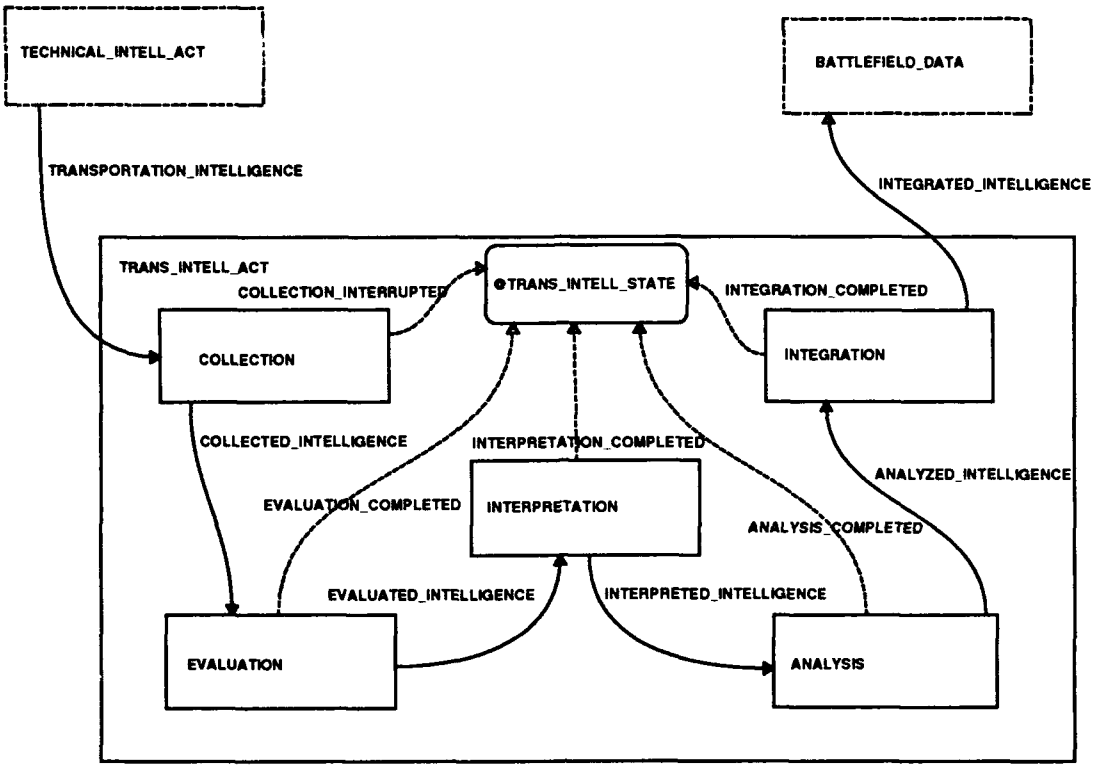


Figure 4-11 Common Intelligence Activities

- Convoy_planning is an aggregate of three features: `convoy_control`, `column_formation`, and `asset`.

The control and internals of these activities will vary depending upon the selection of other features. For example, the selection of the `unit` context feature may mean that convoy_planning will not include `asset`. Control will also vary depending upon the selection among the `control_strategy` features. Section 4.4.2. addresses variability within this control activity.

The functional model does not decompose the operations_activities. The structure of this activity will vary depending on the specific command and control system. The domain analysis only looks at the movement control component of command and control for ATCCS and other related systems requiring movement control. The data passed between activities also exhibits a common structure. Each of the three activities produces and/or uses data created by the other three. As long as this data interface is adhered to, modification of the internal operations of any activity will not affect the others. For example, if convoy_planning is for a `unit` move it will not perform selection of transportation assets. For `logistical` moves, convoy planning must include selection of assets. However, the data convoy_planning exports, consisting of order_of_march and schedules entities, will not change. Section 5.3 will explore this concept as it applies to two actual movement control systems.

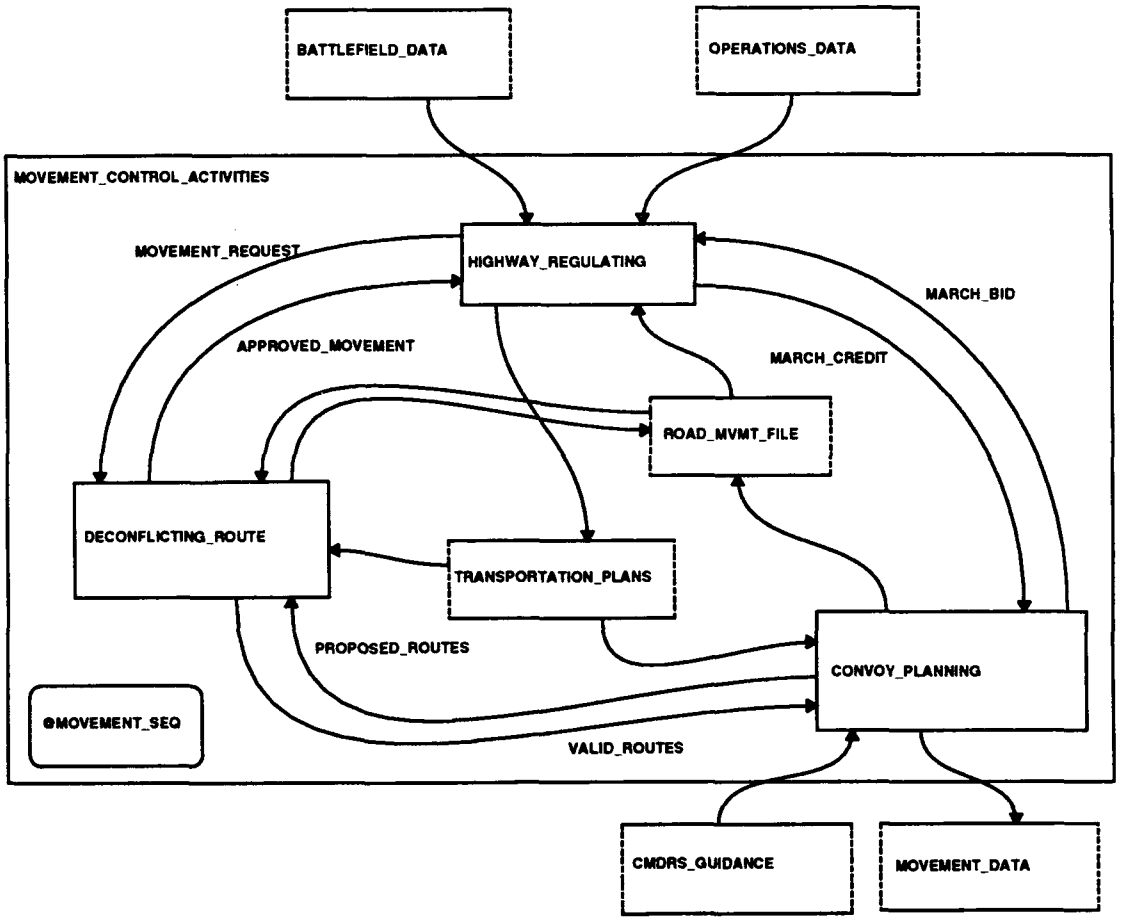


Figure 4-12 Common Movement Control Activity Structure

4.3.4. Domain Dictionary

As stated in Section 2.2., the domain dictionary plays an important role in alleviating misunderstandings among the various groups involved in a domain analysis and between subsequent users of its results. The Army has specific meanings for terms both directly related to movement and for terms used to define organizational levels and structures that are pertinent in understanding movement in military terms. For example:

coil An arrangement of vehicles forming a circle.

This meaning for a common word in the English language is not part of most dictionaries. Alternately, the term movement control is taken from an Army manual [FM 55-10]. Its meaning is more specific and precise than any definition that is derivable from combining the ordinary meanings of these two words.

movement control The planning, routing, scheduling, control, and in-transit visibility of personnel, units, equipment, and supplies moving over lines of communication in accordance with the directives of command planning. It is a continuum involving the synchronization and integration of movement information and programs spanning the strategic, operational, and tactical levels of war. Movement control is guided by a system of balancing requirements against capabilities and allocating resources based on the combat commander's priorities.

The complete listing of the Movement Control Domain Terminology is given in Appendix G.

The Army movement control domain is also rich in acronyms and abbreviations. They play an important role in military communications because their usage can be a significant factor in shrinking the size of messages sent and received between units and in documents generated for widespread use. An example of such an acronym is:

ATCCS Army Tactical Command and Control System

The complete list of movement control domain acronyms is given in Appendix H.

4.4. Validation of the Domain Model

Validation of the domain model products is an important step in the domain analysis process. In order to validate the domain model, it must be possible to specify preexisting or proposed systems using the domain model products. Systems used as inputs in the domain modeling process may be used for validation; but preferably the model developers will test a system not used for developing the model for validation. The features and the functional model are particularly important as they form the basis for the architectural modeling phase. The following two sections discuss the methods used in validating the features and functional model for the movement control domain.

4.4.1. Validation of the Features for Movement Control

Validation of the features was performed by a simple empirical approach that was also used during the feasibility study to validate the window manager features. The representation of the features as seen in Figure 4-5 through Figure 4-8 may be highlighted for those features that are present in each of the two systems analyzed in the greatest detail, AFATDS and DAMMS-R. These systems have significantly different (i.e., non-overlapping) context features that modify more closely related operational features. See Section 3.3 for further information on these two systems.

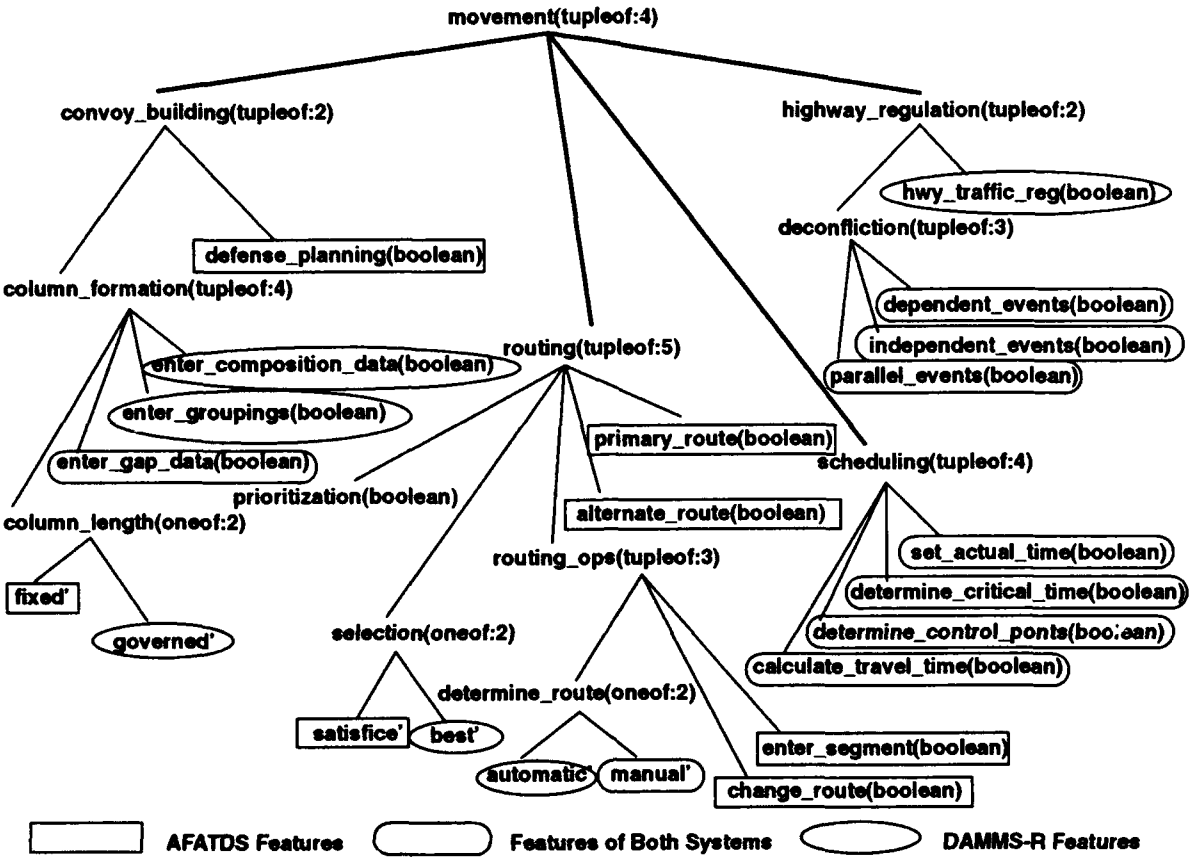


Figure 4-13 Detailed Operational Features for AFATDS and DAMMS-R

Within the scope of the domain analysis, the features highlighted should completely represent the capabilities of the corresponding system or the features are incomplete. A comprehensive feature model is needed to describe a domain with systems that are widely divergent (few features in common) or extremely similar (differing in only a few low-level features). Figure 4-13 and Figure 4-14 highlight the appropriate features of the AFATDS and DAMMS-R systems from the detailed movement operational features (Figure 4-6) and the context features (Figure 4-7).

An even more useful validation of the feature model is to see if it can be used to describe the functionality of a movement control system that was not analyzed during the domain modeling process. The Reserve Component Automation System (RCAS) is to be used by the Army Reserve to support decision making for "all commanders, staff, and functional mangers responsible for Reserve Component Forces." [OO1SRM]. RCAS will support planning and execution of mobilization of reserve units and will also perform routine administrative tasks. The mobilization process consists of bringing together personnel, supplies, and equipment to carry out a specific mission. An important requirement of RCAS in executing mobilization is a process RCAS refers to as "movement execution."

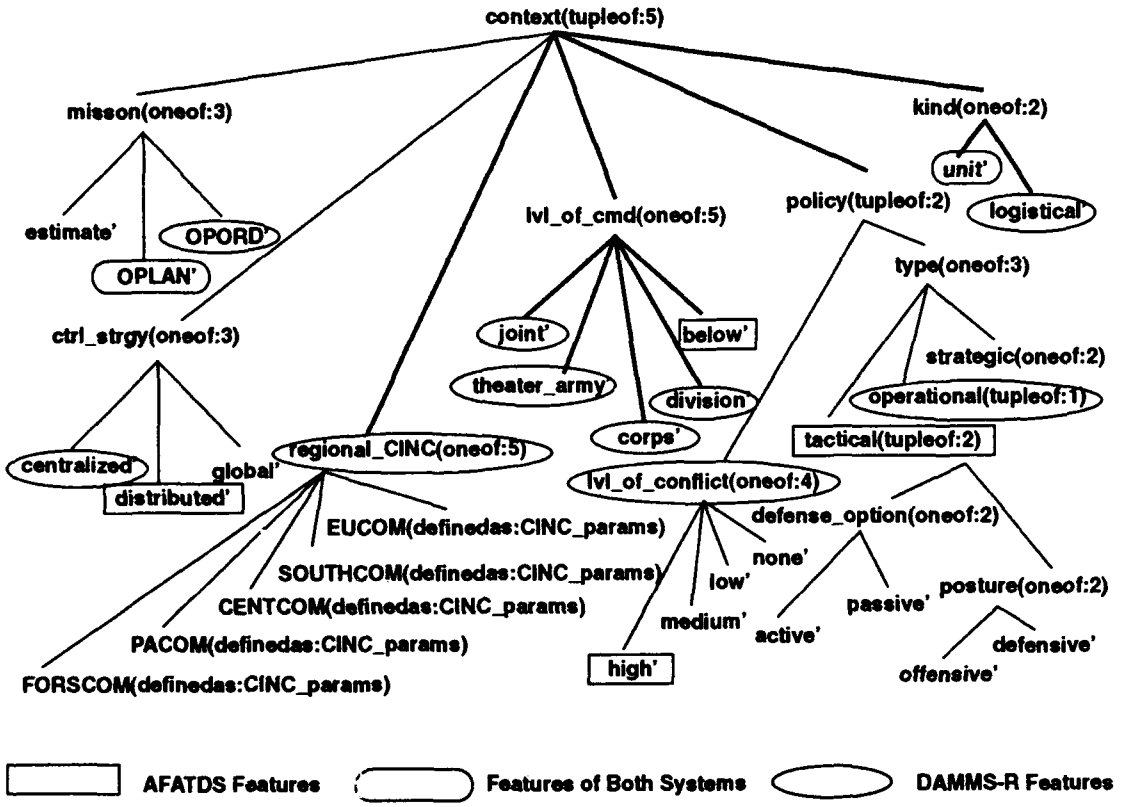


Figure 4-14 Context Features for AFATDS and DAMMS-R

The domain model validation examined this requirement to determine the matching of RCAS needs to features provided in the domain model. Movement control functionality proved to be a vital part of RCAS and serves to validate the model.Figure 4-15 and Figure 4-16 represent the feature mappings of the detailed operational features and the context features, respectively, for the RCAS movement control functionality as described in [OO1SRM]. This comparison reveals two points of validation:

1. The domain model can serve as a reference model for RCAS requirements.
2. The requirements for RCAS very closely parallel those of DAMMS-R.

The information in the domain model (entities, features, and functions) can be used by RCAS developers in their specification activities as they move from function descriptions at a high level to specification. Moreover, following the lead of DAMMS-R, it may be possible to achieve a significant level of reuse between the systems. In this perspective, the domain model and the specifics of DAMMS-R can support for the development of a new system. This is the primary goal of the domain analysis activity.

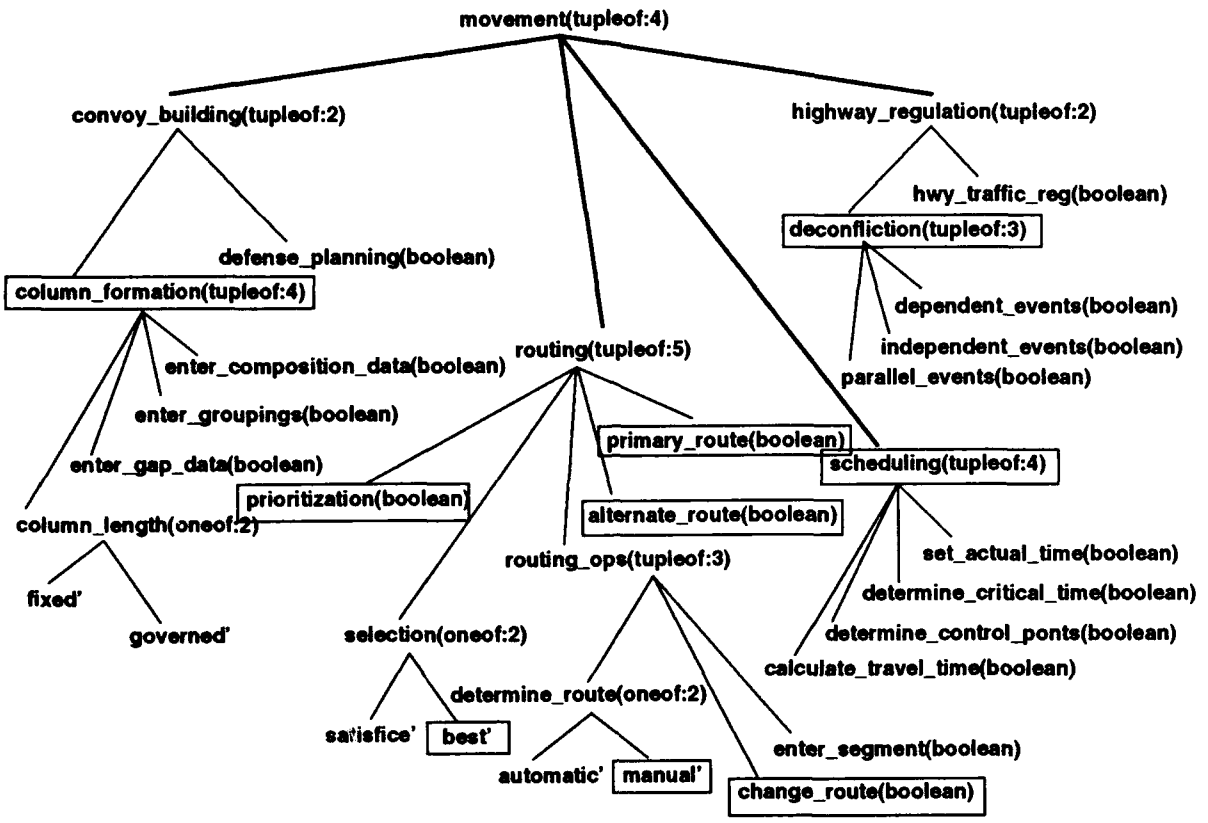


Figure 4-15 Detailed Operational Features for RCAS

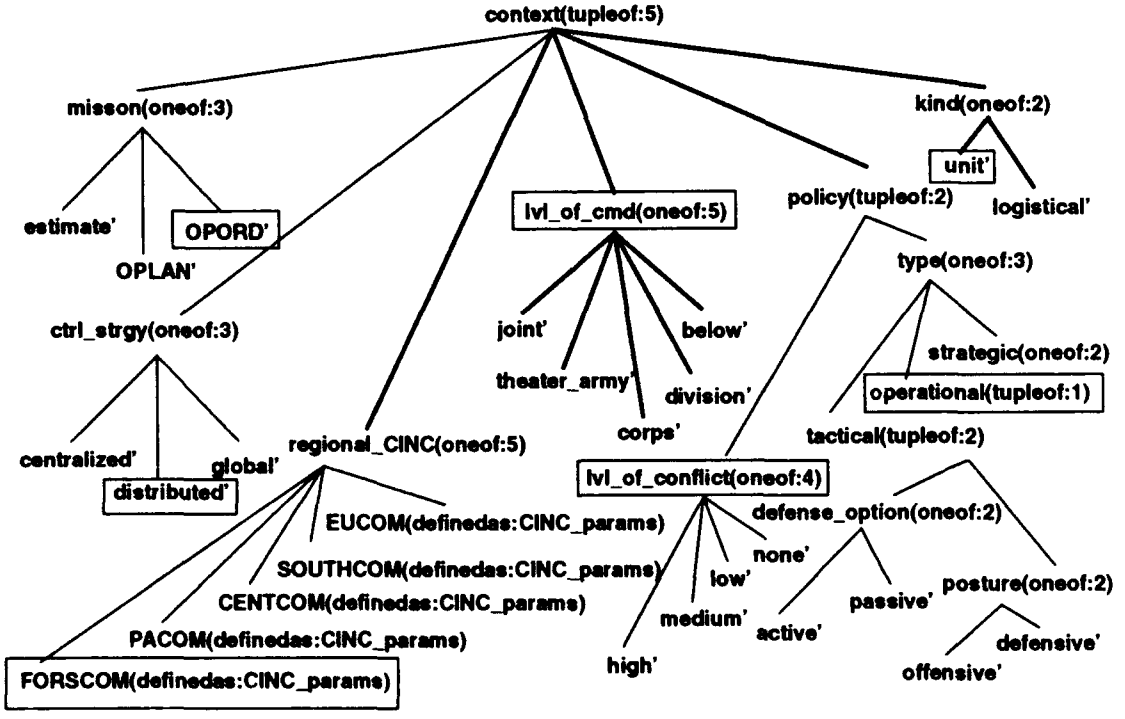


Figure 4-16 Context Features for RCAS

4.4.2. Validation of the Functional Model

Both AFATDS and DAMMS-R share the common functional model presented in Section 4.3.3. These two systems differ, however, in the behavior of their movement_control_activities. This difference is a result of the selection of different `control_strategy` features. As Figure 4-18 shows, AFATDS follows a `distributed control_strategy` while DAMMS-R uses `centralized`. The individual control activities for each `control_strategy` alternative demonstrate the commonality of operations, with differences in specific implementation.

The behavior shown in Figure 4-17 follows the `centralized control_strategy` of DAMMS-R. Within this context, there are three parallel sets of states:

1. Developing_trans_plans controls highway_regulating activities.
2. Deconflicting controls deconflicting_route.
3. Planning_convoy and the other states at the bottom of the figure control convoy_planning activities.

These parallel states will proceed regardless of the internal operations of the activities they control.

For the specific movement_control_activities of AFATDS (Figure 4-18), deconflicting is not a separate state. This is characteristic of `distributed` systems where the entire convoy_planning activity is performed by the unit doing the move. This unit must have the information needed to carry out its own planning because it may be moving under actual combat conditions. This is reflected in the parallelism between developing_trans_plans and planning_convoy on the chart. These states reflect the parallel nature of information gathering and operations planning that is conducted in a combat unit. However, the unit cannot rely on control from a central command to deconflict its move. This activity must be integral to the convoy_planning activity.

4.4.3. Results of Domain Model Validation

The use of actual systems to validate the domain model demonstrates its success in capturing commonality and establishing factors that result in differences among systems. The common structure of the functional model, down to the level of movement_control_activity behavior, shows the activities and control that result from mandatory features. The differing behavior within the movement_control activity shows the effect of different context features to allow for different classes of users and system missions, as well as the differing behaviors among options and alternatives.

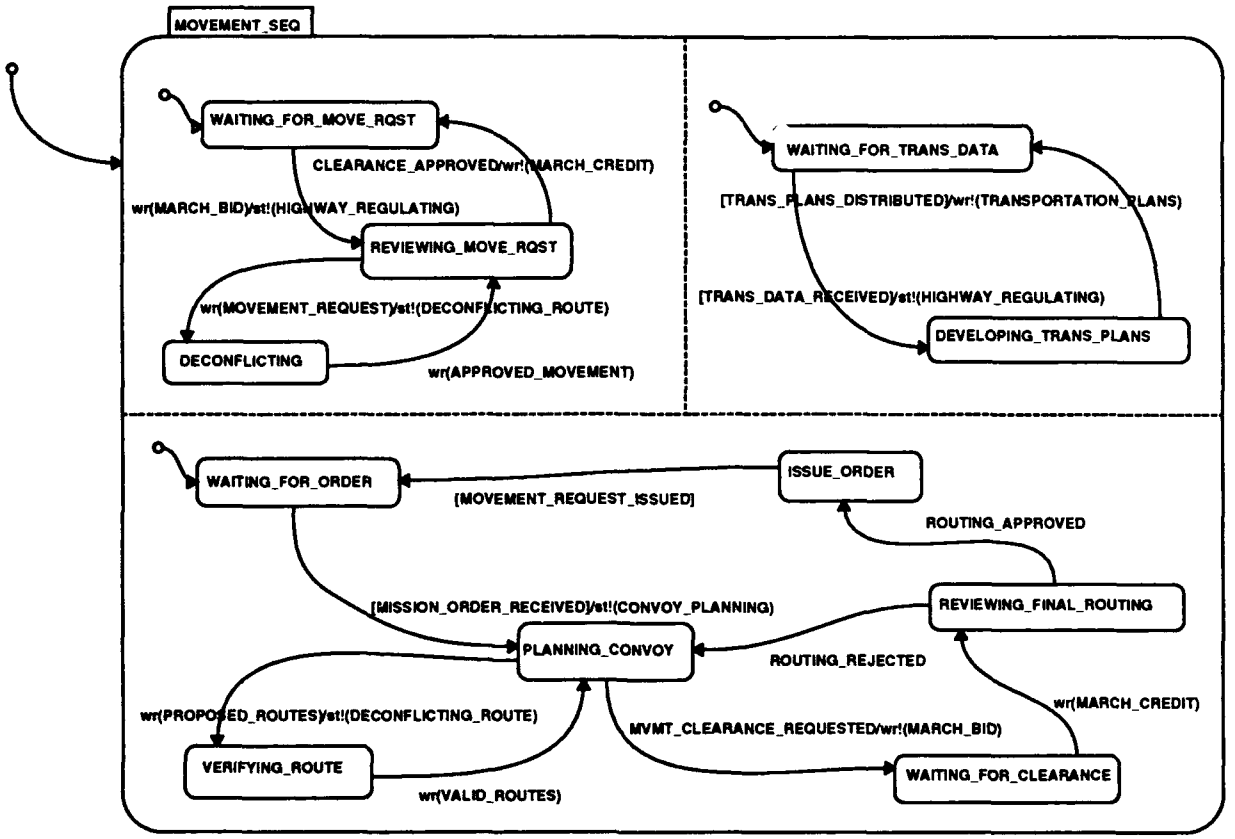


Figure 4-17 Behavior of DAMMS-R Movement_Control_Activities

The next step in validating the domain model is to demonstrate its effectiveness in supporting the development of a new movement control system. Chapter 5 of the report describes the process of using the domain model to build a system and perform prototype testing.

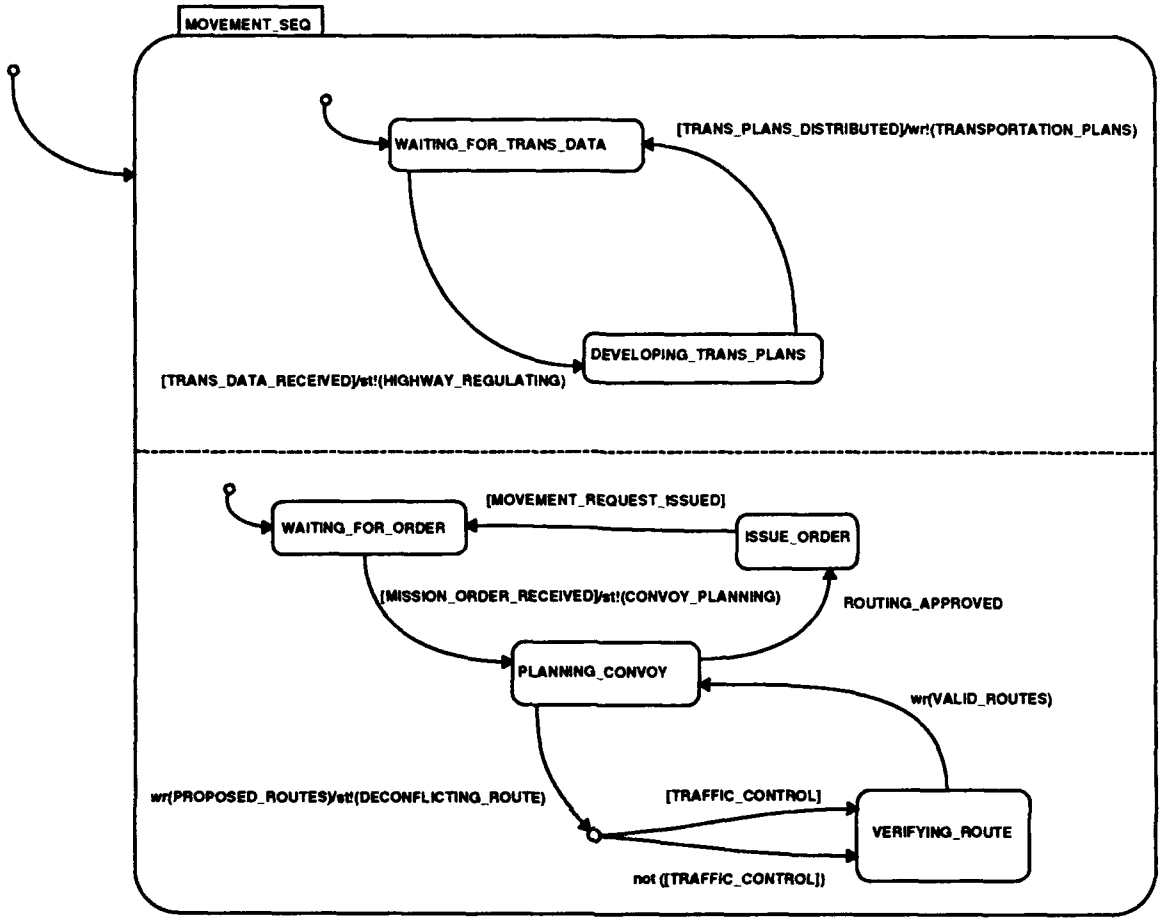


Figure 4-18 Behavior of AFATDS Movement_Control_Activities

5. Application of the Domain Model in System Development

The domain model presented in Chapter 4 forms the basis for software development of a movement control system. By integrating access to the domain model products (the feature, ER, and functional models), the domain modeling tool built using 001 can provide a prototyping capability for movement control. This prototyping system is called the SEI Movement Control Prototyper, or *MoveCon*. The remainder of this chapter will discuss the rationale for building the system, the approach taken in its development, and the use of the system as a means for prototyping various models of movement control systems.

5.1. The Domain Model and Prototyper Capability

A domain modeling tool can be used in the early stages of the software development process to support end users in specifying requirements for new systems. This tool must provide the three views of the FODA domain model. While the information constrained in the model provides enough information to build a system, automatic prototyping gives the user the ability to animate the specification built from a selection of features. Simulation through STATEMATE gives a conceptual animation of the system specification. The simulator approach is not, however, directly tied to the domain model. The functional model in STATEMATE is not integrated with the other views of the domain model. Without integration, it is not possible to test consistency and completeness across modules.

To meet this need for integration, the domain modeling tool must support both domain model creation and model-based prototyping (see Figure 5-1). Using the domain modeling tool, the domain analyst documents the domain model and implements a prototyping capability (labeled Domain Model Prototyper in the figure). The prototyper, as its name implies, allows implementation of a working model of the system under development based on a selection of features. The fidelity of the prototype is a function of:

- The completeness of the domain model (how many of the features within the domain been captured in the model).
- The implementation of code generating capability in the prototyper for those features.

As the domain model matures, more features will be captured in the domain model and, as prototyping verifies parts of the model, the prototyping capability will be increased.

The loops within the figure show the pattern of development of both the model and the prototyper to validate the model. The outermost loop shows the evolution of the model. As the domain analyst captures more domain information, the model is expanded or changed to reflect new domain information. In parallel with domain model development, the domain analyst will construct a working model of the domain via the prototyper. A selection of features captured in the domain model will be built into the prototyper to allow both validation of the domain mod-

el (does the model capture the common features of systems in the domain) and prototyping of a system under development.

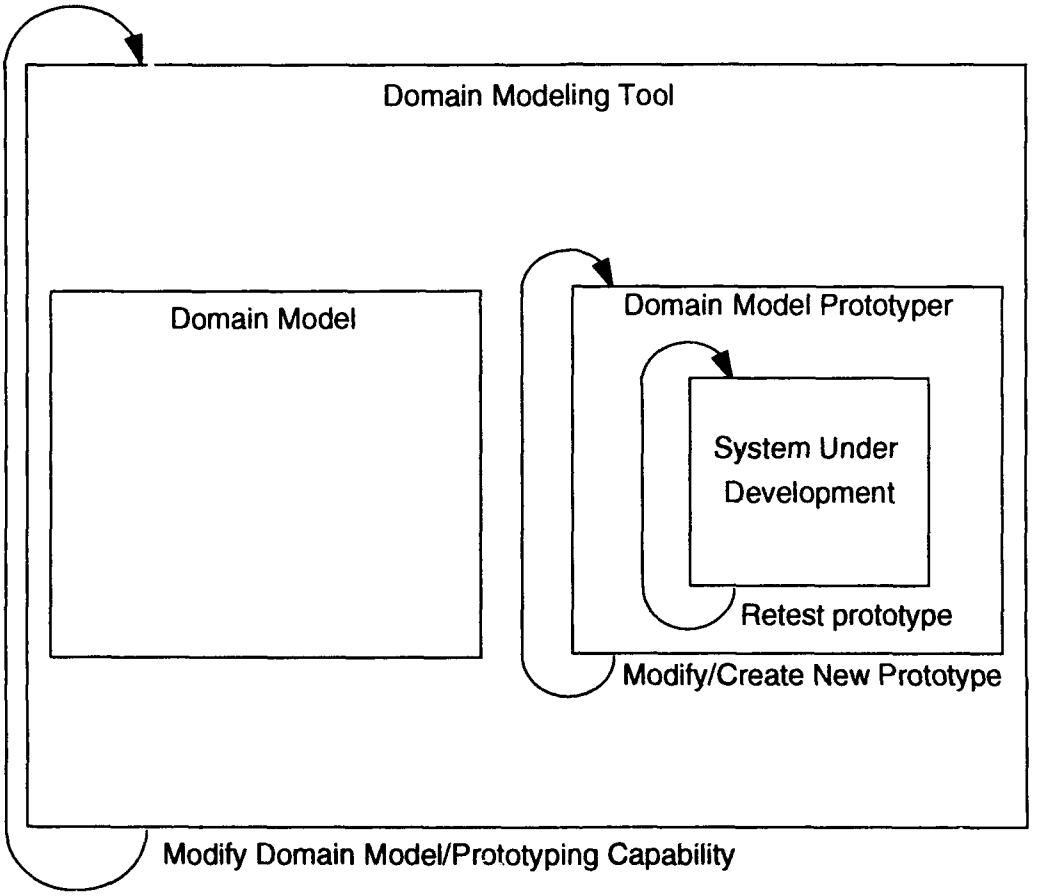


Figure 5-1 Domain Modeling Tool Capability

The loops attached to the prototyper show the ability of the domain modeling tool to test new system capability.

- Given an existing domain model, the prototyper allows selection of features to implement a prototype for a system under development. The user of the prototyper may change features selected for the system to build a new or modified version.

- The prototype may be successively tested. Errors may be traced to incorrect selection of features for the system under development or to incorrect implementation of the features within the prototyper.

The prototyper supports both the domain model developer and user:

- The developer gets feedback on the correctness and completeness of the model.
- The user gets a sense of the capabilities for a new system and the effect of selecting alternative or optional features.

The prototyper directly support the notion of the *binding time* of features, as described in the FODA report:

- *Compile time*. The features implemented in the prototyper that cannot be changed without modifying the domain model or the prototype.
- *Load time*. The features that can be changed whenever the prototyper is used to build a new prototype system or to modify an existing system.
- *Runtime*.The features that can be selected during execution of the system.

The next section describes the use of this approach in building MoveCon, a movement control system prototyper.

5.2. The Movement Control Prototyper (MoveCon)

MoveCon is a prototyping system for implementing those portions of the movement control domain model dealing with convoy planning. The features in the models shown in Figure 5-2 and Figure 5-3 are those available to the user through MoveCon. These features would be included inside the domain model prototyper box of Figure 5-1.

The Army has a need for tools to automate portions of convoy planning both for creating convoys and for determining an appropriate route and schedule for those convoys. In many cases, this function is still performed manually on paper. Several of the steps involved are potentially time-consuming and/or error prone. For example, a combat developer may recognize the need for a new subsystem to automate convoy planning. The three main capabilities of this convoy planning tool are:

1. Selecting vehicles, grouping them as needed into a march column, and providing adequate spacing between vehicles and groups as required.
2. Verifying that one or more selected routes is a valid path of segments from a start point to a release point and that the vehicles in the march column can pass all points on a proposed route. If the vehicles cannot travel the entire route, the route must be rejected.
3. Scheduling the use of the selected routes by calculating the passing and travel times over each segment of the route.

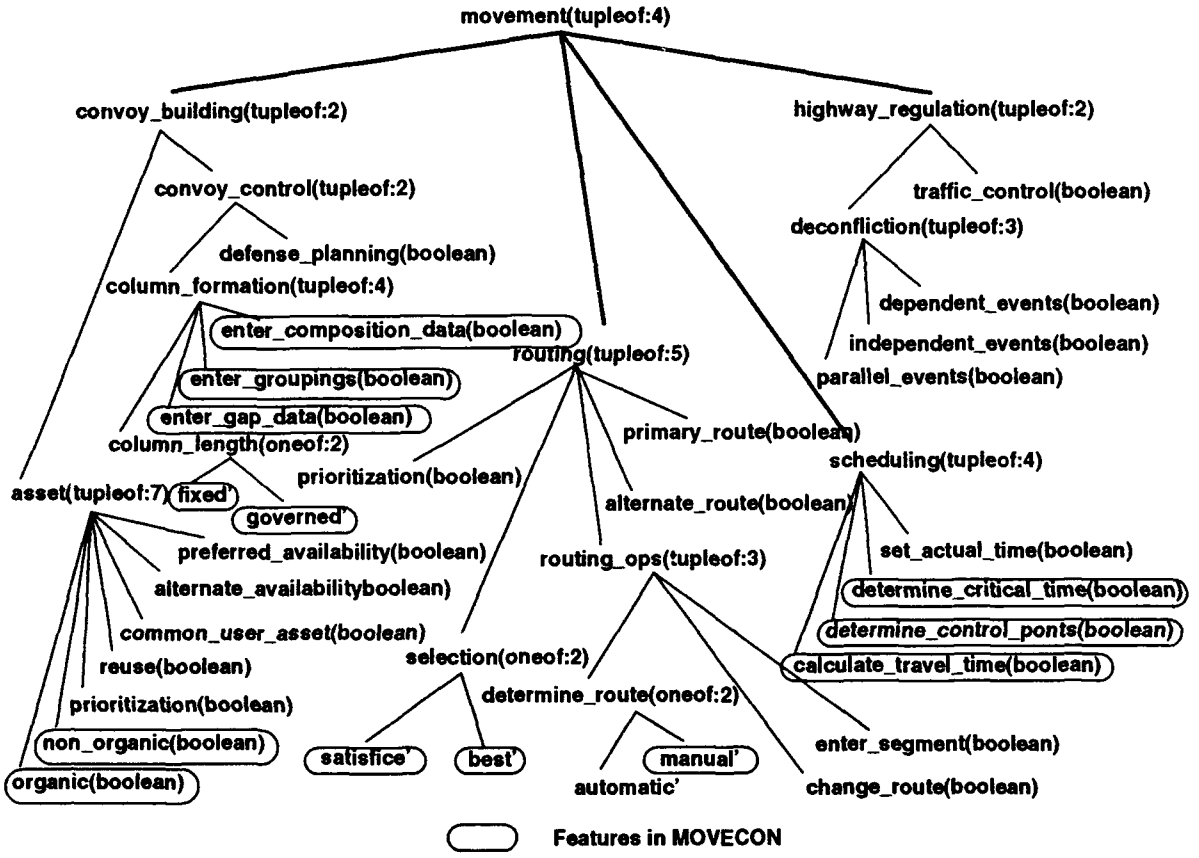


Figure 5-2 The Operational Features Modeled in MoveCon

With minor variations, these three elements (the number and types of vehicles in a convoy, the proposed route, and a schedule for its use) are the essential pieces of information needed by existing or proposed systems. For example, they would support:

- Formulating a Movement Request for AFATDS or the equivalent Convoy Clearance Request for DAMMS-R.
- Performing the convoy planning operations of stand-alone, PC-based systems examined during the domain analysis.

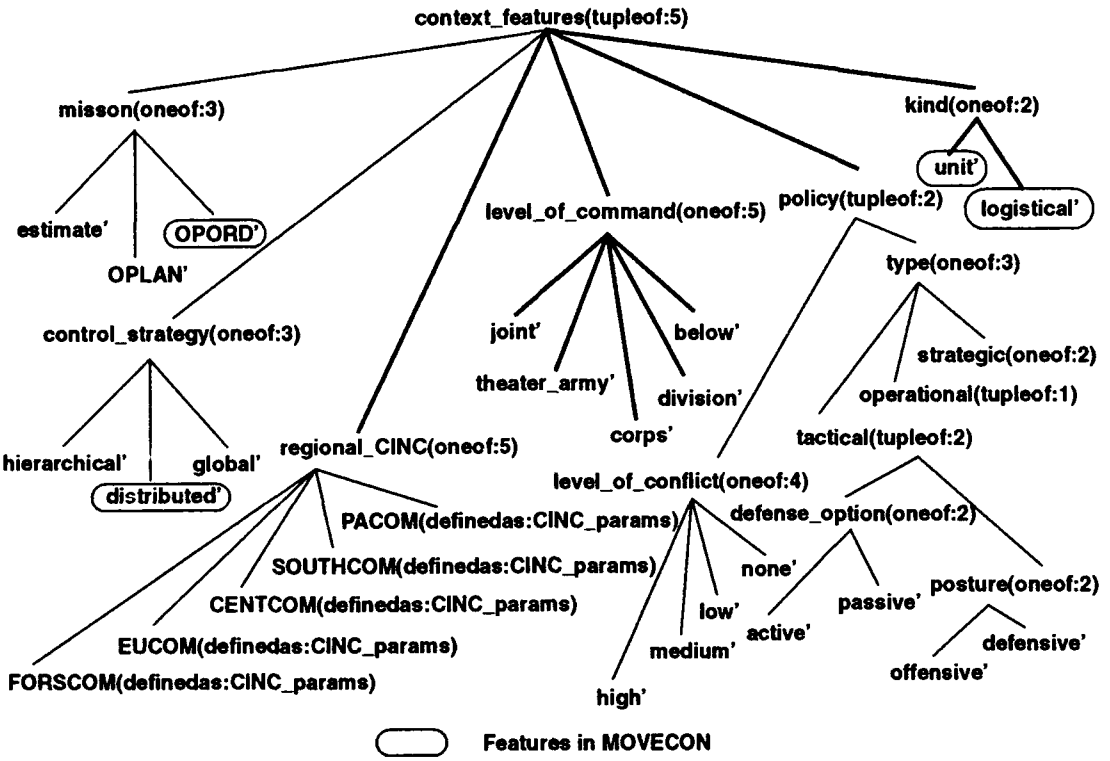


Figure 5-3 The Context Features Modeled in MoveCon

The combat developer can test the system generated by MoveCon as he would any software system. Test data could include convoy composition, route selection, or scheduling constraints. Through successive runs of the prototyper, the combat developer can refine the capabilities of the desired system. These refinements require modifying the prototype system under development, as shown in Figure 5-1, or creating a new prototype. The combat developer may also provide feedback on the prototyping capability to modify the model or the prototyper if the needed capabilities are not part of MoveCon.

5.3. Scenarios to Validate MoveCon

The prototyper has been used to build two convoy planner systems. Figure 5-4 shows the scenarios that exist for each system. Scenario 1 is a simple planner, handling only `unit moves`. Because of the `unit` feature, the composition rules automatically provide both `organic asset` and `governed` columns. Other features of the system for Scenario 1 are not currently selectable in MoveCon. They may be considered *compile-time* features.

Scenario 2 offers more capabilities to the user. During runtime, the user may select either of the alternatives for the `kind` feature. This indicates that for successive convoy building operations, the user of the system can build either `unit` or `logistical` convoys. The runtime

capability for the `kind` feature automatically provides runtime selection for `assets` and `column_length`. Through composition rules, these capabilities must be provided if `logistical` is selected.

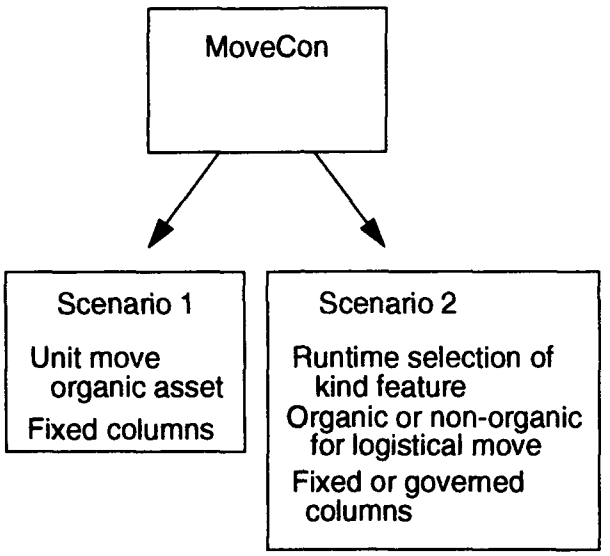


Figure 5-4 Scenarios for Testing the Prototyper

Figure 5-5 and Figure 5-6 show the selection of features for each scenario. Although the features differ in only three places, the resulting differences between the two systems is significant. In Scenario 1, the user has the ability to route and schedule a unit but cannot change unit composition. The vehicles composing the convoy are those of the unit itself and do not change for individual moves. For the second scenario, however, the user has the ability to build the convoy for assets belonging to other units. These non-organic assets will differ depending on the requirements for each move.

Appendix I contains a list of sample vehicle data for both the unit and logistical movement scenarios and a sample road network for building routes.

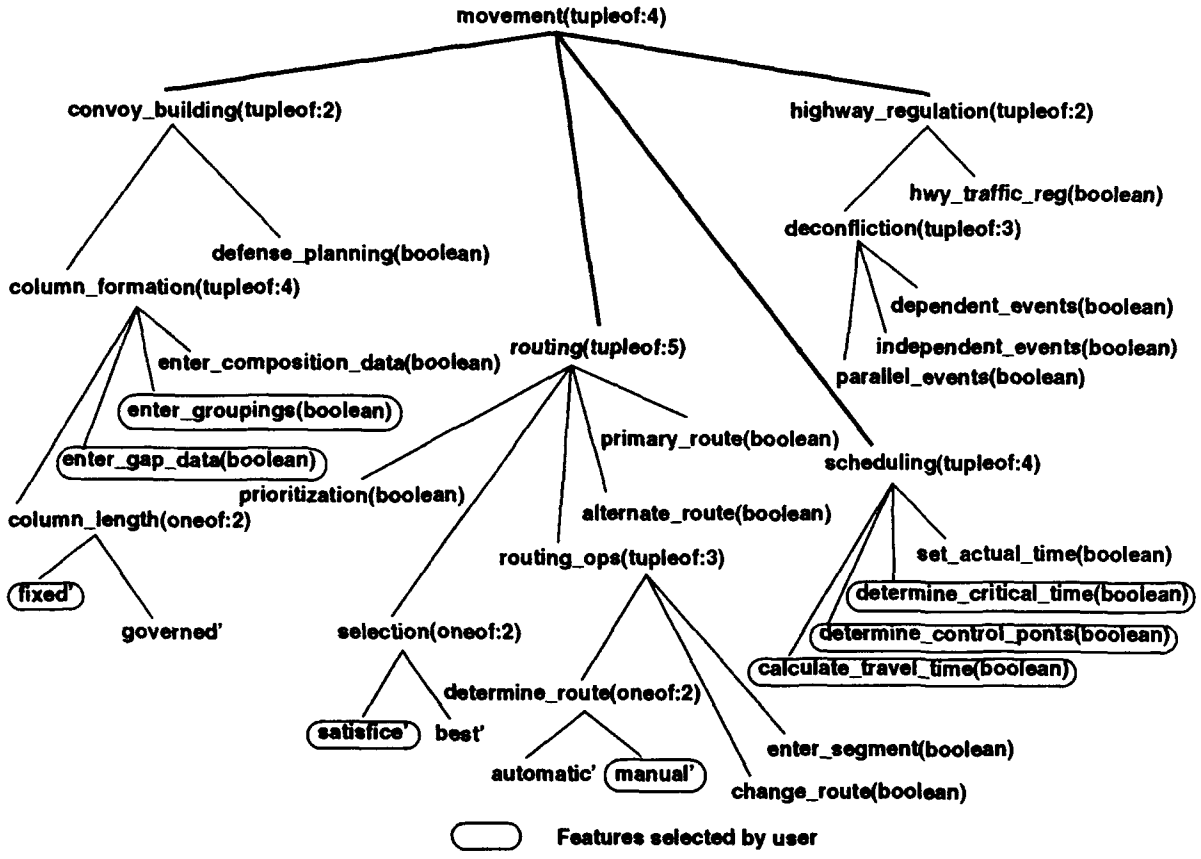


Figure 5-5 Feature Selection for Scenario 1

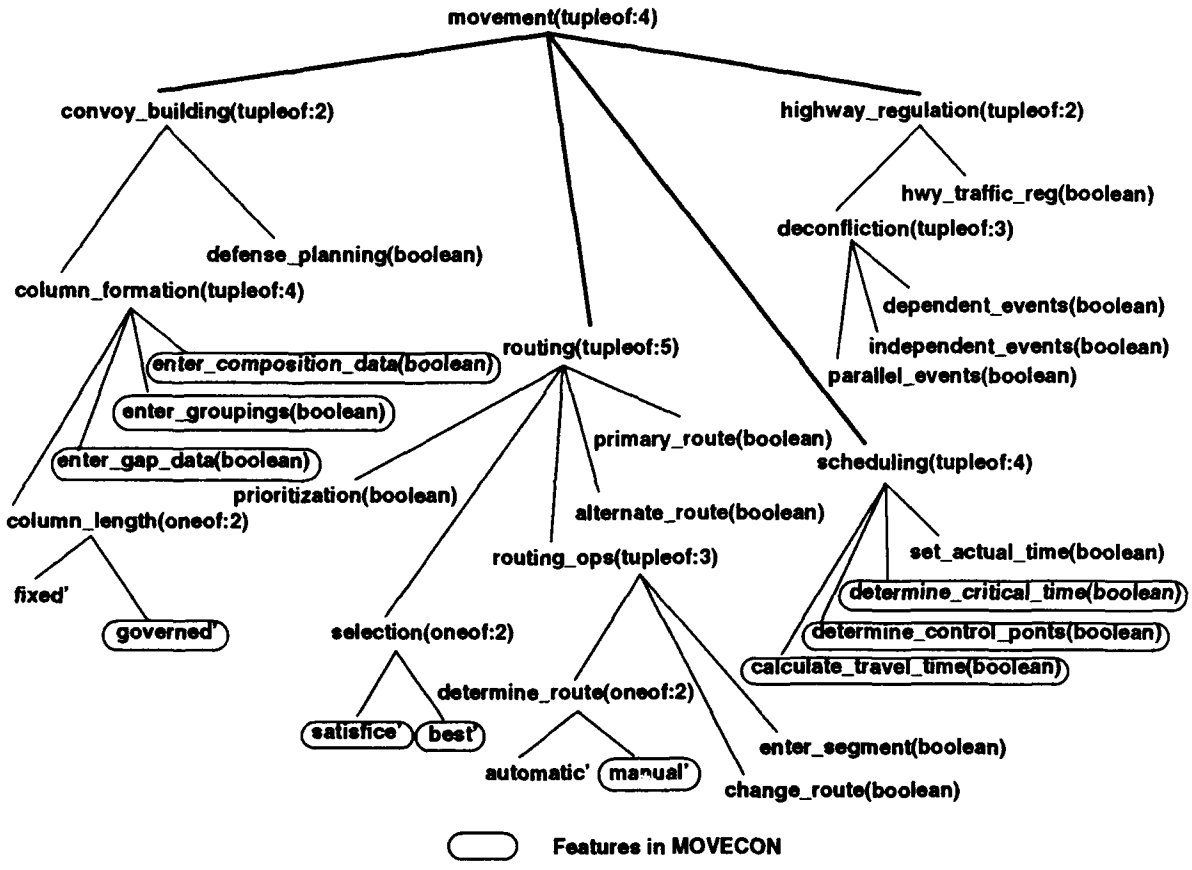


Figure 5-6 Feature Selection for Scenario 2

6. Conclusion

6.1. Lessons Learned

The primary purpose of the Army Movement Control domain analysis was to apply the FODA method within a realistic MCCR domain. An important aspect of this analysis, which differs from that performed under the feasibility study, is that commonality among related systems within the domain was neither well-understood nor well-documented. Domain expertise for movement control did not exist among the domain analysts before the analysis and there was no in-house experience upon which to draw. In addition, before the analysis began there was not an immediately available user of the domain results. In order to obtain confirmation of the model and sample use, a specific user had to be identified.

These factors led to several valuable lessons learned that may be useful to others performing domain analysis.

6.1.1. Clear Definition of Users

As part of context analysis, a domain analyst must identify all users of the domain model. Users could include:

- The combat developers or other system specifiers.
- Software designers or developers.
- Other software systems with which the class of systems under analysis must operate.
- Other domain analysts.

Given this range of users of the domain model, the model must address a range of needs including requirements elicitation, software product implementation, and system interfacing. Representatives of each of the user groups must be involved in the development and review of the model and order to assure its usability.

6.1.2. Early Identification of Domain Experts

During context analysis, the domain analyst must identify the appropriate domain experts and other sources of information. For the movement control domain, this includes combat developers from several Army TRADOC organizations concerned with movement as well as specific Army manuals and procedures. Learning from the domain experts, the domain analysts become "experts" as well, but are constantly measuring their knowledge against the true domain expertise. Attendance at training courses within the domain is also an appropriate means of building in-house expertise. Effectively working with domain experts is the best means to achieve adoption of the domain model by potential users.

6.1.3. Need for Enactment of Model

During the feasibility study, Statemate simulation was used to demonstrate the capabilities of a system specified using feature selection. This was effective in a simple domain (window managers), where potential users have a good idea of how such systems work. For the movement control domain, potential users must be convinced that the model can be effectively used to specify a new system. The use of a modeling tool that can both capture the domain model and produce prototype code to simulate a system based on feature selection is one approach that can secure user acceptance.

6.1.4. Establishing a Community of Interest

During later stages of the analysis, it is very useful to bring together domain experts from different organizations for joint activities. These meetings can be organized around discussions of the domain model, its application, or their own work in progress. For example, it was discovered during this domain analysis that groups as diverse as university researchers, an AI firm, and the Army Transportation School have a significant overlap of interest. The domain analysis process brought them together to share their interests. Once a domain model is in place, it is necessary to maintain contact with all current and potential users of the model to assure its successful application and evolution.

6.1.5. Establishing Domain Expert Support

A domain analysis effort must have formalized support agreements established between domain analysts. During the movement control analysis, participation from domain experts came through selling the idea and hoping experts would become interested enough to work with the analyst. Although the group managed to elicit adequate support to build the model, the process was time-consuming and should not be used for subsequent efforts. In the future, once experts are identified, a memorandum of understanding should be developed between the two parties that solidifies the domain expert involvement throughout the project.

6.2. Recommendations for Future Research

6.2.1. Observations Leading to Research Recommendation

Within the Army there seems to be a lack of communication among developers of related software systems. Little detailed information is exchanged from command to command. This means that related software systems that perform primarily the same operations are often developed independently. Domain analysis helps improve communication. Through analysis of the specific domain for which the software is targeted, information about related systems is exposed.

The development of the Department of the Army Movement Management System (DAMMS-R) illustrates the benefits of improved communication among developers. DAMMS-R is being developed to support movement management, transportation operations, and common user transportation asset control functions within any theater of operations. Movement control plays

a significant role in the operation of the DAMMS-R software. After taking a closer look at the requirements for DAMMS-R, it was determined that the DAMMS-R requirements and movement control requirements for the ATCCS system are very similar. Recognizing these similarities, the domain analysts orchestrated interaction between the interested parties for both systems. As a result, the possibility of a joint development effort was discovered. The domain analysis effort is now attempting to provide products to both the DAMMS-R and ATCCS projects.

The domain analysis also discovered that the ALBE-GIS, part of the Geographical Information System for the ATCCS system, provides tactical decision aids (TDA) that may support movement control. This means that part of the movement control software may already exist.

Without analyzing the software domain prior to development, these reuse opportunities would have probably been overlooked.

6.2.2. Proposed Directions

The Army should apply the movement control domain model and SEI software architecture technology to implement movement control software for ATCCS. This software will interface with the common ATCCS support software (CASS) architectures. Software development will use the domain model and analysis method together with proven methods used by the SEI Software Architecture Engineering Project.

The task will be done jointly with interested Army parties who have participated in the analysis. The project will also receive domain information from several ongoing prototyping efforts (Army Transportation School, Carnegie Group, AFATDS) that are not currently developed on the CASS configuration. The capabilities of these prototypes will help refine the movement control domain model and will be reflected in the movement control architecture.

This task will also support the CASS group in determining effective methods for applying the results of the domain analysis in the creation of reusable software. The SEI will look at the implications of designing movement control packages for reuse in Ada.

The products produced from this work will be:

- A fully developed movement control domain model with automated tool support.
- An architecture to support development of reusable movement control software.
- Various reusable Ada packages which will serve as examples for CASS and guidelines for implementing reusable software with Ada 9X.

Domain analysis tutorial
Domain model-based Ada package
Ada reuse handbook
Domain analysis handbook

Table 6 Tentative Schedule for Future Research

6.2.3. Premise for Research Recommendations

It is recommended that responsible Army organizations support these proposed directions. Both the method and system developers can receive the benefits of a joint effort.

- Refinement of the FODA method by incorporating SAE work.
- Joint development of two major Army systems needing primarily the same capability.
- Application of FODA to new domains.

Joint support means that the Army would receive a movement control software architecture, reusable components, and the technology associated with the development. This software could be used on subsequent software developments as a by-product of ongoing research, without the need for initiating a new development.There are three primary benefits associated with this joint interaction:

1. Since both systems primarily use the same software, a joint development venture will be more cost effective for both commands, leading to overall savings for the Army.
2. Since DAMMS-R interfaces with ATCCS, many of the problems normally associated with software interoperability and interfacing can be avoided.
3. If these systems are developed from a common software approach, the user interfaces will be similar. For example, soldiers who are trained to interact with movement control software (regardless of the systems incorporating the common application) can be taught from this standard interface.

References

[ACAM86] *Army Command and Management, Theory and Practice.* US Army War College, Carlisle Barracks, PA, August 19, 1986.

[BODIN81] Lawerence Bodin, Bruce Golden, Arjang Assad, & Mark Ball. *The State Of the Art in the Routing and Scheduling of Vehicles and Crews.* Technical Report UMTA-URT-41-81-1, US Department of Transportation, Washington, DC, September 1981.

[FM 34-1] Army Field Manual FM 34-1, *Intelligence and Electronic Warfare Operations.* Headquarters, Department of the Army, Washington, DC, July 1987.

[FM 34-3] Army Field Manual FM 34-3, *Intelligence Analysis.* Headquarters, Department of the Army, Washington, DC, January 1986.

[FM 55-10] Army Field Manual FM 55-10, *Movement Control in a Theater of Operations.* Headquarters, Department of the Army, Washington, DC, November 1990.

[FM 101-5] Army Field Manual FM 101-5, *Staff Organization and Operations.* Headquarters, Department of the Army, Washington, DC, May 1984.

[GOMAA84] Hassan Gomaa. "A Software Design Method for Real-Time Systems." *Communications of the ACM*, Vol. 27(9): 938-949, September 1984.

[HAMIL86] Margaret H. Hamilton. "Zero-defect software: the elusive goal." *IEEE Spectrum*, Vol. 23(3):48-xx, March 1986.

[HAMIL90] Margaret H. Hamilton & William R. Hackler. "001: A Rapid Development Approach for Rapid Prototyping Based on a System that Support Its Own Life Cycle." *Proceedings of the First International Workshop on Rapid System Prototyping*, IEEE Computer Society Press, June 1990, pg. 46-62.

[MURPH90] Erin E. Murphy. "Software R & D: from an art to a science." *IEEE Spectrum*, Vol 27(10): 44-46, October 1990.

[OO1SRM] The 001™ Tool Suite, *System Reference Manual*, Version 2D.10 Hamilton Technologies, Inc., Cambridge, MA, May 1991.

[RCAS91] ASDM 18-J04-HVR-XXX-FD, *Reserve Component Automation System (RCAS) Functional Description*, Appendix F, Mobilization Planning, Part A, User Functional Requirements RCAS Program Management Office, Newington, VA, February 1991.

[SEI88] Kenneth Lee, Michael Rissman, Richard D'Ippolito, Charles Plinta, & Roger Van Scoy.
An OOD Paradigm for Flight Simulators, 2nd Edition (CMU/SEI-88-TR-30, ADA204849).
Software Engineering Institute, Pittsburgh, PA, September 1988.

[SEI90a] Kyo C. Kang, Sholom G. Cohen, James A. Hess, William E. Novak, & A. Spencer Peterson.
Feature-Oriented Domain Analysis (FODA) Feasibility Study (CMU/SEI-90-TR-21, ADA235785).
Software Engineering Institute, Pittsburgh, PA, November 1990.

[SEI90b] Michael M. Rissman, Richard D'Ippolito, Ken Lee & Jeffrey Stewart.
Definition of Engineering Requirements for AFECO, Lessons Learned from Flight Simulators (CMU/SEI-90-SR-25).
Software Engineering Institute, Pittsburgh, PA, December 1990.

[SEI91a] A. Spencer Peterson & Sholom G. Cohen.
A Context Analysis of the Movement Control Domain for the Army Tactical Command and Control System (ATCCS) (CMU/SEI-91-SR-3).
Software Engineering Institute, Pittsburgh, PA, April 1991.

[SEI91b] Robert W. Krut Jr. & David P. Wood.
Evaluation of Process Modeling Improvements (CMU/SEI-91-TR-5).
Software Engineering Institute, Pittsburgh, PA, April 1991.

[STALL87] Richard Stallman.
GNU Emacs Manual, Sixth Edition, Version 18.
Free Software Foundation, Cambridge, MA, March 1987.

[YOUNG89] Douglas A. Young.
X Windows System - Programming And Applications with Xt.
Prentice-Hall, Englewood Cliffs, NJ, 1989.

[ZIMM80] Hubert Zimmerman.
OSI Reference Model - The ISO Model for Architecture of Open Systems Interconnection.
IEEE Transactions on Communications, Vol. 28(4): 425-432, April 1980.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release Distribution Unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S CMU/SEI-91-TR-28	5. MONITORING ORGANIZATION REPORT NUMBER(S) ESD-TR-91-28
6a. NAME OF PERFORMING ORGANIZATION Software Engineering Institute 6b. OFFICE SYMBOL (if applicable) SEI	7a. NAME OF MONITORING ORGANIZATION SEI Joint Program Office
6c. ADDRESS (City, State and ZIP Code) Carnegie Mellon University Pittsburgh PA 15213	7b. ADDRESS (City, State and ZIP Code) ESD/AVS Hanscom Air Force Base, MA 01731
8a. NAME OFFUNDING/SPONSORING ORGANIZATION SEI Joint Program Office 8b. OFFICE SYMBOL (if applicable) ESD/AVS	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F1962890C0003
8c. ADDRESS (City, State and ZIP Code) Carnegie Mellon University Pittsburgh PA 15213	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO 63756E PROJECT NO. N/A TASK NO N/A WORK UNIT NO. N/A
11. TITLE (Include Security Classification) Application of Feature-Oriented Domain Analysis to the Army Movement Control Domain	
12. PERSONAL AUTHOR(S) Sholom G. Cohen, Jay L. Stanley, Jr., A. Spencer Peterson, and Robert W. Krut, Jr.	
13a. TYPE OF REPORT Final 13b. TIME COVERED FROM TO	14. DATE OF REPORT (Yr., Mo., Day) June 1992 15. PAGE COUNT 75
16. SUPPLEMENTARY NOTATION	

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse of necessary and identify by block number)
FIELD	GROUP	SUB. GR.	common Army terminology and requirements domain analysis tool support movement control

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This report documents an analysis of the army movement control domain performed by the Software Engineering Institute (SEI) and a team of movement control experts from the Army. This report includes common terminology and requirements extracted from Army doctrine, experts in the field, and movement control systems. The report also describes the potential for prototyping of systems using domain analysis products and the tool support needed.

(please turn over)

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED SAME AS RPTDTIC USERS ■	21. ABSTRACT SECURITY CLASSIFICATION Unclassified, Unlimited Distribution
22a. NAME OF RESPONSIBLE INDIVIDUAL John S. Herman, Capt, USAF	22b. TELEPHONE NUMBER (Include Area Code) (412) 268-7631 22c. OFFICE SYMBOL ESD/AVS (SEI)

ABSTRACT —continued from page one, block 19